

UNIVERSIDADE FEDERAL DE OURO PRETO  
INSTITUTO DE CIÊNCIAS EXATAS E BIOLÓGICAS  
DEPARTAMENTO DE COMPUTAÇÃO

CLARA LÓRIS DE SALES GOMES  
Orientador: Prof. Dr. Reinaldo Silva Fortes

**RECSYSEXP 2:  
APRIMORANDO O FRAMEWORK DE IMPLEMENTAÇÃO E  
VALIDAÇÃO EM SISTEMAS DE RECOMENDAÇÃO**

Ouro Preto, MG  
2024

UNIVERSIDADE FEDERAL DE OURO PRETO  
INSTITUTO DE CIÊNCIAS EXATAS E BIOLÓGICAS  
DEPARTAMENTO DE COMPUTAÇÃO

CLARA LÓRIS DE SALES GOMES

**RECSYSEXP 2:**  
**APRIMORANDO O FRAMEWORK DE IMPLEMENTAÇÃO E VALIDAÇÃO EM**  
**SISTEMAS DE RECOMENDAÇÃO**

Monografia apresentada ao Curso de Ciência da Computação da Universidade Federal de Ouro Preto como parte dos requisitos necessários para a obtenção do grau de Bacharel em Ciência da Computação.

**Orientador:** Prof. Dr. Reinaldo Silva Fortes

Ouro Preto, MG  
2024



## FOLHA DE APROVAÇÃO

Clara Lóris de Sales Gomes

**RecSysExp 2: aprimorando o framework de implementação e validação em sistemas de recomendação**

Monografia apresentada ao Curso de Ciência da Computação da Universidade Federal de Ouro Preto como requisito parcial para obtenção do título de Bacharel em Ciência da Computação

Aprovada em 22 de Fevereiro de 2024.

Membros da banca:

Reinaldo Silva Fortes (Orientador) - Doutor - Universidade Federal de Ouro Preto  
Anderson Almeida Ferreira (Examinador) - Doutor - Universidade Federal de Ouro Preto  
Bráulio Miranda Veloso (Examinador) - Mestre - Universidade Federal de Ouro Preto

Reinaldo Silva Fortes, Orientador do trabalho, aprovou a versão final e autorizou seu depósito na Biblioteca Digital de Trabalhos de Conclusão de Curso da UFOP em 22/02/2024.



Documento assinado eletronicamente por **Reinaldo Silva Fortes, PROFESSOR DE MAGISTERIO SUPERIOR**, em 22/02/2024, às 09:36, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do [Decreto nº 8.539, de 8 de outubro de 2015](#).



A autenticidade deste documento pode ser conferida no site [http://sei.ufop.br/sei/controlador\\_externo.php?acao=documento\\_conferir&id\\_orgao\\_acesso\\_externo=0](http://sei.ufop.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0), informando o código verificador **0667217** e o código CRC **F49D9103**.

*Dedico este trabalho com imensa gratidão aos meus queridos pais, Cátia Sales e Ricardo Antunes, e à minha irmã, Luna. Obrigada por todo apoio e incentivo, sem vocês nada disso seria possível.*

# Agradecimentos

Agradeço primeiramente a Deus, cuja presença iluminou cada passo desta jornada. Agradeço aos meus pais, Cátia e Ricardo, por sua confiança e apoio inabaláveis ao longo da minha trajetória. Um agradecimento especial estende-se à minha irmã, Luna, e ao João Vicente, por me escutarem e me darem tanto apoio nessa jornada.

Expresso minha sincera gratidão à minha Tia Nem, cujas contribuições ao longo da minha vida são inestimáveis. Às minhas avós, que, embora não estejam mais fisicamente presentes, permanecem vivas em meu coração durante esta caminhada. Agradeço também aos meus tios, tias, primos e primas, assim como aos queridos nenéns Ivan, Derick e Helena, que me trouxeram imensa felicidade, mesmo à distância.

Meus agradecimentos se estendem ao professor Reinaldo Fortes, cuja disponibilidade e orientação foram de valor inestimável nessa etapa. Ao Lucas Natali, agradeço por sua contribuição fundamental para a realização deste trabalho.

À UFOP e ao DECOM, expresso minha profunda gratidão pelo aprendizado proporcionado. Não posso deixar de mencionar a importância de todos os amigos que encontrei em Ouro Preto, em especial, minhas amigas que sempre me apoiaram, as quais conquistei na República Rosa Xiclete, onde compartilhei momentos significativos ao longo da graduação. Agradeço também aos meus amigos do curso, por toda contribuição e apoio.

# Resumo

O constante aumento de dados disponíveis gerou a necessidade de Sistemas de Recomendação capazes de filtrar informações relevantes aos usuários. No entanto, a busca por conteúdo verdadeiramente significativo para os usuários, muitas vezes com necessidades e expectativas diversas, demanda a incorporação de métricas além da acurácia, como a novidade, a diversidade ou a serendipidade. Para lidar com isso, torna-se necessário a aplicação de técnicas de otimização multiobjetivo. O *framework* **RecSysExp**, desenvolvido por Natali (2023), foi criado com o intuito de simplificar a realização de experimentos em recomendação, abrangendo todas as etapas do processo. No entanto, o *framework* apresenta limitações ao não contemplar recomendações multiobjetivo e métricas de avaliação que não estejam relacionadas com a medida de acurácia, o que fundamenta a expansão proposta por este trabalho. O objetivo central deste trabalho é enriquecer o **RecSysExp** para suprir essas limitações. Isso envolve a criação de uma infraestrutura que viabilize a integração de métodos para a implementação de Sistemas de Recomendação multiobjetivo, e a incorporação de métricas de avaliação que considerem a novidade das recomendações, tornando-o mais abrangente. Ao expandir o **RecSysExp** este trabalho visa ampliar os cenários de uso do *framework*. Foram conduzidos experimentos computacionais, a fim de demonstrar a eficácia das melhorias introduzidas no *framework* **RecSysExp**. O experimento foi estruturado para testar a capacidade do sistema aprimorado em lidar com recomendações multiobjetivo.

**Palavras-chave:** Sistemas de Recomendação. *Framework*. Recomendação multiobjetivo.

# Abstract

The constant increase in available data has generated the need for Recommendation Systems capable of filtering relevant information for users. However, the search for truly meaningful content for users, often with diverse needs and expectations, requires the incorporation of metrics beyond accuracy, such as novelty, diversity, or serendipity. To address this, the application of multi-objective optimization techniques becomes necessary. The *framework RecSysExp*, developed by Natali (2023), was created with the intention of simplifying the conduct of experiments in recommendation, covering all stages of the process. However, the *framework* has limitations by not considering multi-objective recommendations and evaluation metrics unrelated to accuracy, which justifies the expansion proposed by this work. The central objective of this work is to enrich **RecSysExp** to address these limitations. This involves the inclusion of methods that allow the implementation of multi-objective Recommendation Systems and the incorporation of evaluation metrics that consider the novelty of recommendations, making it more comprehensive. By expanding **RecSysExp**, this work aims to broaden the use cases of the framework. Computational experiments were conducted to demonstrate the effectiveness of the improvements introduced in the *framework RecSysExp*. The experiment was structured to test the enhanced system's ability to handle multi-objective recommendations.

**Keywords:** Recommendation Systems. Framework. Multi-objective

# Lista de Ilustrações

Figura 2.1 – Frente de Pareto (VILLA; LABAYRADE, 2011). . . . .	9
Figura 2.2 – Arquitetura Geral RecSysExp (NATALI, 2023). . . . .	17
Figura 2.3 – Arquitetura Xperimentor (DIAS, 2019). . . . .	20
Figura 3.1 – Representação da arquitetura do módulo de métricas. . . . .	26
Figura 3.2 – Representação da arquitetura do módulo de recomendação (NATALI, 2023). . . . .	28
Figura 3.3 – Arquitetura do módulo incluindo suporte para métodos multiobjetivo. . . . .	29
Figura 4.1 – Evolução hipervolume STREAM - POP100. . . . .	37
Figura 4.2 – Evolução hipervolume HR - POP100. . . . .	38
Figura 4.3 – Representação decisão da melhor solução com diferentes pesos. . . . .	40

# Lista de Tabelas

Tabela 4.1 – Variações Experimentos. . . . .	36
Tabela 4.2 – Intervalo de confiança novidade - HR - POP100. . . . .	42
Tabela 4.3 – Intervalo de confiança novidade - HR - POP500. . . . .	42
Tabela 4.4 – Intervalo de confiança novidade - STREAM - POP100. . . . .	42
Tabela 4.5 – Intervalo de confiança novidade - STREAM - POP500. . . . .	42
Tabela 4.6 – Intervalo de Confiança Acurácia - HR - POP100. . . . .	43
Tabela 4.7 – Intervalo de Confiança Acurácia - HR - POP500. . . . .	43
Tabela 4.8 – Intervalo de confiança acurácia - STREAM - POP100. . . . .	43
Tabela 4.9 – Intervalo de confiança acurácia - STREAM - POP500. . . . .	44

# Lista de Abreviaturas e Siglas

ABNT	Associação Brasileira de Normas Técnicas
DECOM	Departamento de Computação
MOEA	<i>Multi-objective Evolutionary Algorithm</i>
EPC	<i>Expected Popularity Complement</i>
FD	<i>Free Discovery</i>
EILD	<i>Expected Intra-List Distance</i>
RMSE	<i>Root Mean Squared Error</i>
MAE	<i>Mean Absolute Error</i>
NDCG	<i>Normalized Discounted Cumulative Gain</i>
XML	<i>Extensible Markup Language</i>
NSGA-II	<i>Non-dominated Sorting Genetic Algorithm II</i>
NSGA-III	<i>Non-dominated Sorting Genetic Algorithm III</i>
MOEA/D	<i>Multi-Objective Evolutionary Algorithm based on Decomposition</i>
AGE-MOEA	<i>Adaptive Genetic Evolution Multi-Objective Evolutionary Algorithm</i>
DCG	<i>Discounted Cumulative Gain</i>
IDCG	<i>Ideal Discounted Cumulative Gain</i>
NDCG	<i>Normalized Discounted Cumulative Gain</i>
TPC	<i>Two Point Crossover</i>
SBC	<i>Simulated Binary Crossover</i>
PM	<i>Polynomial Mutation</i>
GM	<i>Gaussian Mutation</i>

# Sumário

<b>1</b>	<b>Introdução</b>	<b>1</b>
1.1	Justificativa	2
1.2	Objetivos	2
1.3	Organização do Trabalho	3
<b>2</b>	<b>Revisão Bibliográfica</b>	<b>4</b>
2.1	Fundamentação Teórica	4
2.1.1	Sistemas de Recomendação	4
2.1.1.1	Filtragem baseada em conteúdo	5
2.1.1.2	Filtragem colaborativa	5
2.1.1.3	Filtragem híbrida	6
2.1.1.4	Filtragem multiobjetivo	8
2.1.2	Avaliação de Sistemas de Recomendação	12
2.2	Trabalhos Relacionados	15
2.2.1	RecSysExp	15
2.2.2	RecMetrics	18
2.2.3	Xperimentor	19
2.2.4	RankSys	21
2.2.5	Elliot	21
2.2.6	DaysiRec 2.0	22
2.2.7	Pymoo: Multi-objective Optimization in Python	23
<b>3</b>	<b>Desenvolvimento</b>	<b>25</b>
3.1	Inclusão da métrica de novidade	25
3.2	Inclusão de métodos multiobjetivo	27
3.2.1	Otimização Multiobjetivo	29
3.2.1.1	Definição do problema	29
3.2.1.2	Etapas de busca	30
3.2.1.3	Tomada de decisão	32
3.2.2	Predição	33
<b>4</b>	<b>Experimentos</b>	<b>35</b>
4.1	Base de dados	35
4.2	Configurações dos experimentos	35
4.3	Resultados	36
4.3.1	Otimização	37
4.3.2	Decisão Melhor Solução	39
4.3.3	Predição	41
<b>5</b>	<b>Conclusões</b>	<b>45</b>

**Referências** . . . . . 47

# 1 Introdução

Os Sistemas de Recomendação surgiram como solução para lidar com a sobrecarga de informações. Com o aumento exponencial de dados, os usuários se deparam com um bombardeio de dados, nem sempre relevantes (SUN et al., 2019). Nesse contexto, os Sistemas de Recomendação desempenham um papel crucial, podendo gerar listas de itens que atendam às necessidades do usuário (ZHENG; WANG, 2021).

A área de Sistemas de Recomendação destaca-se como uma ferramenta fundamental para auxiliar os usuários em suas tomadas de decisões. Eles oferecem sugestões personalizadas com base nas preferências e comportamentos individuais. O avanço da tecnologia e a crescente disponibilidade de informações tornam esses sistemas ainda mais relevantes, por serem amplamente aplicados em diversos domínios, como plataformas de streaming, sites de comércio eletrônico e redes sociais. Empresas renomadas, como Netflix e Spotify, utilizam esses sistemas para melhorar a experiência do usuário, fornecendo resultados mais relevantes e personalizados (ZHENG; WANG, 2022).

Segundo Zheng e Wang (2021), os Sistemas de Recomendação foram inicialmente projetados com foco na otimização de um único objetivo, historicamente centrado na avaliação da acurácia de classificação. No entanto, atualmente, compreende-se que o sucesso de um sistema de recomendação depende de outros fatores essenciais (RIBEIRO et al., 2015). É importante perceber que as métricas padrão de acurácia nem sempre correspondem às recomendações mais úteis para os usuários (MCNEE; RIEDL; KONSTAN, 2006b).

Os usuários possuem preferências e necessidades diversas. Em uma plataforma de streaming, por exemplo, um usuário pode não desejar receber recomendações previsíveis constantemente. Em vez disso, ele pode valorizar a diversidade e a descoberta de conteúdos novos e interessantes, que seria a novidade (ZHENG; WANG, 2021). No entanto, essa pluralidade de objetivos pode levar a conflitos, criando o desafio do problema multiobjetivo.

Conforme mencionado por Ribeiro et al. (2015), buscar maximizar a novidade e a diversidade sem considerar a acurácia seria insensato. O objetivo, nesse caso, é encontrar um equilíbrio entre essas métricas ou melhorar outras métricas sem comprometer totalmente a acurácia das recomendações (ZHENG; WANG, 2021). Portanto, é fundamental encontrar uma abordagem que considere diferentes fatores e necessidades dos usuários para fornecer recomendações personalizadas e aprimoradas. Dessa forma, os Sistemas de Recomendação estão em constante progresso para lidar com essa complexidade e atender às necessidades diversificadas dos usuários, considerando múltiplos objetivos simultaneamente.

Nesse sentido, torna-se essencial a exploração de Sistemas de Recomendação multiobjetivo, que possam conciliar e equilibrar diversos critérios de recomendação. Com o crescente

número de publicações sobre Sistemas de Recomendação e a constante proposta de novos algoritmos de recomendação a cada ano (SUN et al., 2022), é de fundamental importância o desenvolvimento de sistemas que facilitem a execução de experimentos em recomendação. Essas ferramentas são essenciais para conduzir pesquisas e testes eficientemente, possibilitando um avanço mais significativo nessa área em constante evolução.

Este trabalho dá continuidade ao *framework* de experimentação de Sistemas de Recomendação, previamente desenvolvido por Natali (2023). A meta é estender esse *framework*, chamado **RecSysExp**, para oferecer suporte a Sistemas de Recomendação multiobjetivo e incluir uma métrica de novidade, para que seja possível aplicar o multiobjetivo.

As Seções 1.1, 1.2 e 1.3 a seguir, abordam, respectivamente, a motivação por trás deste trabalho, os objetivos esperados para o desenvolvimento do **RecSysExp** e a organização deste trabalho.

## 1.1 Justificativa

Com o contínuo aumento do volume de dados disponíveis, a criação de Sistemas de Recomendação se torna uma necessidade crucial para filtrar e disponibilizar informações relevantes aos usuários. Nesse contexto, é de fundamental importância o desenvolvimento de sistemas que facilitem a execução de experimentos em recomendação, por serem essenciais para a condução de pesquisas e testes eficientemente.

O *framework* **RecSysExp**, proposto por Natali (2023), oferece a capacidade abrangente de conduzir experimentos completos, desde o pré-processamento até a avaliação e visualização de resultados. No entanto, é importante notar que o *framework* apresenta algumas limitações nos cenários experimentais que cobre, como a ausência de métricas que considerem a novidade das recomendações, restringindo a análise de critérios além da acurácia. A falta de uma estrutura e métodos multiobjetivo também impede a realização de experimentos que buscam avaliar simultaneamente estas diferentes métricas.

Diante dessas limitações, surge a necessidade de aprimorar o *framework* **RecSysExp**, possibilitando a realização de experimentos mais diversificados em sistemas de recomendação. Esta melhoria justifica-se pelo fato de que a inclusão de métodos multiobjetivo e métricas que considerem a novidade das recomendações permitirá que o *framework* seja aplicado a uma gama mais ampla de cenários experimentais.

## 1.2 Objetivos

Este trabalho visa ampliar o *framework* **RecSysExp**, tornando-o mais abrangente em diferentes cenários de recomendação. Isso será alcançado através da inclusão de uma métrica específica de novidade, assim como a adaptação para métodos multiobjetivo.

Para atingir o objetivo principal, os seguintes objetivos específicos são definidos:

1. Incluir Métrica de Avaliação de Novidade: Integrar ao *framework* uma métrica que avalie a novidade nas recomendações;
2. Incluir Métodos Multiobjetivo: Adicionar ao *framework* métodos multiobjetivo para experimentos de recomendação que envolvam diferentes objetivos simultaneamente.

### 1.3 Organização do Trabalho

Este trabalho é estruturado da seguinte forma: no Capítulo 2, é realizada uma revisão bibliográfica, que engloba tanto a fundamentação teórica quanto uma análise de trabalhos relacionados. O Capítulo 3 detalha a metodologia adotada para atingir os objetivos propostos. O Capítulo 4, apresenta os experimentos e resultados obtidos. Finalmente, no Capítulo 5, são apresentadas as considerações finais, juntamente com as perspectivas para o futuro desenvolvimento deste trabalho.

## 2 Revisão Bibliográfica

O campo dos Sistemas de Recomendação tem sido objeto de ampla exploração e estudo, impulsionado pelo crescente interesse em fornecer recomendações personalizadas aos usuários. Nesse contexto, será apresentado, na Seção 2.1, uma fundamentação teórica dos principais conceitos necessários para compreender este trabalho. Em seguida, na Seção 2.2, será realizada uma revisão bibliográfica de *frameworks* e bibliotecas relacionados ao escopo deste estudo.

### 2.1 Fundamentação Teórica

Esta seção apresentará fundamentos importantes para compreender este trabalho. A Subseção 2.1.1 aprofundará a noção de Sistemas de Recomendação e suas metodologias. Posteriormente, na Subseção 2.1.2, elucidaremos alguns conceitos sobre avaliações em Sistemas de Recomendação.

#### 2.1.1 Sistemas de Recomendação

De acordo com Ricci, Rokach e Shapira (2011), Sistemas de Recomendação desempenham o papel de oferecer sugestões de itens aos usuários, auxiliando-os em processos de tomada de decisão, como seleção de músicas ou filmes, por exemplo. O indivíduo para o qual a recomendação é direcionada é denominado usuário, enquanto o elemento sendo sugerido é referido como um item, de acordo com Aggarwal (2016). Essas sugestões visam aprimorar a experiência do usuário, proporcionando alternativas personalizadas conforme suas preferências individuais.

Os Sistemas de Recomendação são alimentados por informações que capturam as preferências dos usuários em relação a uma variedade de itens. Esses dados podem ser obtidos de forma explícita, por meio de avaliações diretas fornecidas pelos usuários, ou de maneira implícita, monitorando comportamentos como os sites visitados ou os itens adquiridos, por exemplo (AGGARWAL, 2016; BOBADILLA et al., 2013).

A integração de várias fontes de informação é um aspecto crucial nos Sistemas de Recomendação, visando fornecer previsões e recomendações equilibradas e precisas, conforme ressaltado por Aggarwal (2016). Para alcançar tal objetivo, são empregadas diversas abordagens. As técnicas de recomendação conhecidas possuem vantagens e desvantagens, havendo momentos em que a combinação de diferentes técnicas é necessária para alcançar o resultado desejado, como mencionado por Burke (2002). Nas próximas subseções, exploraremos algumas dessas abordagens: filtragem baseada em conteúdo, filtragem colaborativa, filtragem híbrida e filtragem multiobjetivo.

### 2.1.1.1 Filtragem baseada em conteúdo

Os Sistemas de Recomendação baseados em conteúdo visam sugerir itens semelhantes àqueles pelos quais um usuário demonstrou interesse anteriormente. Essa abordagem compara as preferências e interesses de um usuário, com os atributos de um item específico para descobrir os itens mais relevantes para esse usuário. Esse processo permite apresentar novos itens de interesse ao usuário, aumentando seu engajamento (NABIZADEH et al., 2013; LOPS; GEMMIS; SEMERARO, 2011).

De acordo com Lops, Gemmis e Semeraro (2011), esses sistemas, que se baseiam na filtragem baseada em conteúdo, operam identificando afinidades entre os itens candidatos e aqueles que já agradaram ao usuário anteriormente. Ao analisar padrões e preferências históricas, é possível obter informações valiosas que guiam escolhas futuras. A relevância desse método é ampliada pela consideração das correlações entre itens, resultando em recomendações mais precisas e relevantes (AGGARWAL, 2016).

A filtragem baseada em conteúdo oferece várias vantagens notáveis. Ela é independente de outros usuários, o que permite fornecer recomendações precisas sem depender de interações sociais. Além disso, esses sistemas podem incorporar rapidamente novos itens no conjunto de dados, desde que seus atributos sejam disponíveis. No entanto, é importante observar que esses sistemas têm suas limitações. Sua eficácia depende da qualidade das informações dos itens, o que pode exigir esforços significativos para coletar e pré-processar essas informações. Além disso, os algoritmos baseados em conteúdo tendem a ficar limitados ao escopo temático existente, o que pode dificultar a geração de recomendações surpreendentes e inovadoras (NABIZADEH et al., 2013).

### 2.1.1.2 Filtragem colaborativa

De acordo com Rashid et al. (2002), a filtragem colaborativa é uma técnica que os Sistemas de Recomendação utilizam para sugerir itens com base no feedback dos usuários. Os usuários expressam suas opiniões ao avaliar os itens com os quais interagiram em um determinado domínio de escolhas. O sistema de recomendação, então, compara essas avaliações com as de outros usuários, encontra usuários similares usando critérios de similaridade e propõe itens que esses usuários similares gostaram em ocasiões anteriores.

As principais vantagens da Filtragem Colaborativa, segundo Fortes (2022), são que ela pode funcionar efetivamente sem depender de informações detalhadas sobre atributos específicos. Isso a torna flexível e aplicável a uma ampla variedade de domínios. Além disso, essa abordagem pode surpreender os usuários ao apresentar itens que eles podem não ter descoberto por conta própria. Isso é possível ao considerar os gostos e padrões de usuários similares, permitindo que os usuários explorem novos itens que provavelmente serão de seu interesse.

É fundamental entender que a Filtragem Colaborativa estima as preferências dos usuários

por itens baseando-se em padrões compartilhados entre usuários ou itens similares. Ela sugere itens considerando o que usuários com gostos parecidos já consumiram, supondo que o usuário em questão também apreciará esses itens. No entanto, a eficácia dessa abordagem depende da disponibilidade de informações sobre usuários, itens e avaliações, o que pode ser problemático em casos de escassez de dados (SCHAFER et al., 2007).

No contexto de novos usuários ou itens, emerge um desafio notável para essa abordagem, chamado *Cold Start*, em que o sistema depara-se com a carência de histórico de avaliações, limitando a criação de sugestões individualizadas. Uma abordagem comum consiste em sugerir itens populares aos recém-chegados. Contudo, essa tática revela-se insuficiente para gerar recomendações verdadeiramente customizadas. Para obter previsões personalizadas, é fundamental que o sistema obtenha informações sobre as preferências do recém-chegado. A abordagem mais direta para essa questão consiste em solicitar avaliações diretas, apresentando diversos itens ao usuário com o intuito de colher informações e, assim, ajustar as futuras recomendações conforme suas preferências individuais (RASHID et al., 2002; NABIZADEH et al., 2013; SUN et al., 2019).

### 2.1.1.3 Filtragem híbrida

De acordo com Burke (2002), a abordagem híbrida em Sistemas de Recomendação visa aprimorar a eficácia das recomendações através da combinação de estratégias distintas. Essa metodologia envolve a fusão de duas ou mais técnicas de filtragem, como a filtragem colaborativa e a filtragem baseada em conteúdo. Essa sinergia permite que as vantagens de cada abordagem se complementem, ao mesmo tempo, em que suas desvantagens são atenuadas.

Em meio à busca por um desempenho otimizado, a comunidade de pesquisa de Sistemas de Recomendação se depara com o desafio de harmonizar as características únicas de cada técnica. Os sistemas híbridos, nesse contexto, se destacam ao unir duas ou mais abordagens de recomendação, permitindo uma melhoria significativa no desempenho sem sucumbir às limitações inerentes de uma única técnica (NABIZADEH et al., 2013).

A filtragem híbrida visa superar alguns obstáculos, dentre eles o *Cold Start* que acontece em filtragem colaborativa. Segundo Çano e Morisio (2019), para superar esse desafio, os sistemas híbridos integram a filtragem colaborativa com métodos como mineração de regras de associação ou outras abordagens matemáticas que investigam as características dos itens.

Além disso, a escassez de dados é outra problemática pontuada por Çano e Morisio (2019), recorrente nos Sistemas de Recomendação. A ausência de avaliações disponíveis compromete a qualidade das sugestões oferecidas. Para lidar com esse desafio, as abordagens híbridas integram diversas técnicas de manipulação de matrizes juntamente com estratégias básicas de recomendação. Além disso, essas abordagens também exploram informações adicionais, como as características individuais dos itens, a fim de contornar a limitação dos dados.

A abordagem híbrida, conforme apresentada por [Shah et al. \(2017\)](#), apresenta uma diversidade de estratégias, sendo algumas delas exemplificadas a seguir. A estratégia de *switching* possibilita a alternância entre diferentes técnicas, adaptando-se a critérios específicos para escolher a abordagem mais adequada. Já na abordagem *mixed*, são geradas listas de recomendações por diversas abordagens e então são combinadas gerando uma única lista. Na estratégia *weighted*, adotada neste trabalho, as técnicas de recomendação contribuem para calcular o valor de uma sugestão por meio de ponderações.

Conforme descrito por [Fortes \(2022\)](#), a estratégia *weighted* implica na combinação de resultados provenientes de várias técnicas. Um método de aprendizagem é empregado para determinar o peso ( $w$ ) atribuído a cada *feature* ( $f$ ), resultando no score ( $S$ ) para um par de item e usuário definido por:

$$S = w_1 \cdot f_1 + w_2 \cdot f_2 + \dots + w_n \cdot f_n \quad (2.1)$$

Nessa equação, as *features*,  $f_1, f_2, \dots, f_n$ , são pontuações geradas por algoritmos externos para pares específicos de item e usuário, e  $w_1, w_2, \dots, w_n$  são os pesos calculados para cada *feature*, onde  $n$  é o número total de *features*.

A utilização de ponderação possibilita que cada método tenha uma influência proporcional no processo de geração de recomendações. A combinação ponderada dos resultados visa otimizar a precisão e relevância das sugestões finais. Dessa forma, a hibridização ponderada oferece uma abordagem flexível para otimizar a colaboração entre diferentes métodos, aprimorando, assim, a eficácia global do sistema de recomendação.

Outra estratégia híbrida que será utilizada, proposta por [Bao, Bergman e Thompson \(2009\)](#), é o *Stacking Recommendation Engines with Additional Meta-features* (STREAM). Essa abordagem incorpora o uso de *meta-features*, que são características implícitas nos dados de entrada que podem exercer influência sobre os algoritmos, conforme apontado por [Fortes \(2022\)](#). Elas têm a capacidade de capturar detalhes que não são diretamente observáveis nos dados primários. A inclusão dessas informações complementares enriquece a análise, permitindo abordagens personalizadas que elevam a precisão das recomendações ([ADOMAVICIUS; ZHANG, 2012](#)).

O STREAM opera em duas etapas: a primeira gera *scores* relacionado ao conjunto de algoritmos selecionados e valores associados às *meta-features* que foram aplicadas. Enquanto a segunda etapa constrói as *features* pela união dos *scores* e *meta-features*. Após a geração das *features*, o processo de aplicação de pesos segue a mesma lógica da hibridização ponderada explicada anteriormente.

### 2.1.1.4 Filtragem multiobjetivo

Na atualidade, somos constantemente confrontados com problemas multiobjetivo em nosso dia a dia. Tomar decisões que busquem conciliar diferentes metas e desejos é uma realidade comum. Um exemplo clássico seria em uma plataforma de delivery. Os usuários podem ter preferência por estabelecimentos que ofereçam frete gratuito, ao mesmo tempo em que valorizam o tempo de entrega e a proximidade do restaurante. Porém, esses objetivos podem entrar em conflito, tornando a decisão mais complexa.

Nesse sentido, para lidar com vários objetivos simultaneamente e encontrar um equilíbrio entre eles, pode-se potencialmente empregar a otimização multiobjetivo. Nesse contexto é importante entender alguns conceitos, como o princípio de dominância, que possibilitará o entendimento sobre como se definem as melhores soluções de um problema multiobjetivo. Dominância é definida por [Ferreira \(2018\)](#) como o seguinte:

Seja  $s_1$  uma solução representada por um vetor de funções objetivos  $f_1 = \{f_{1_1}, \dots, f_{1_m}\}$ , e  $s_2$  outra solução representada por um vetor de funções objetivos  $f_2 = \{f_{2_1}, \dots, f_{2_m}\}$ . A dominância de  $s_1$  sobre  $s_2$  ocorre quando  $s_1$  é igual ou superior a  $s_2$  em todos os objetivos e estritamente superior em pelo menos um deles. Considerando que valores menores nas funções objetivos indicam melhor desempenho, estabelecemos as condições de dominância:

- Para todo índice  $i$ ,  $1 \leq i \leq m$ , temos que  $f_{1_i} \leq f_{2_i}$ .
- Existe pelo menos um índice  $k$ ,  $1 \leq k \leq m$ , tal que  $f_{1_k} < f_{2_k}$ .

Em termos formais:

$$\forall i, 1 \leq i \leq m : f_{1_i} \leq f_{2_i}$$

$$\exists k, 1 \leq k \leq m : f_{1_k} < f_{2_k}$$

A definição de dominância é uma ferramenta essencialmente empregada em problemas de otimização multiobjetivo, permitindo a comparação de soluções com múltiplos critérios e a identificação daquelas que se destacam como superiores ou dominantes em relação às demais.

Outro conceito importante são os relacionados ao Pareto-ótimo. Segundo [Lücken, Barán e Brizuela \(2014\)](#), uma solução é considerada Pareto-ótima quando não é passível de ser dominada por qualquer outra solução pertencente ao conjunto  $X$  de soluções. O conjunto resultante de todas as soluções não dominadas é denominado conjunto ótimo de Pareto. A representação gráfica desse conjunto, conhecida como a Fronteira de Pareto, é composta por todas as soluções Pareto-ótimas. Tal representação pode ser observada na [Figura 2.1](#), que oferece uma valiosa

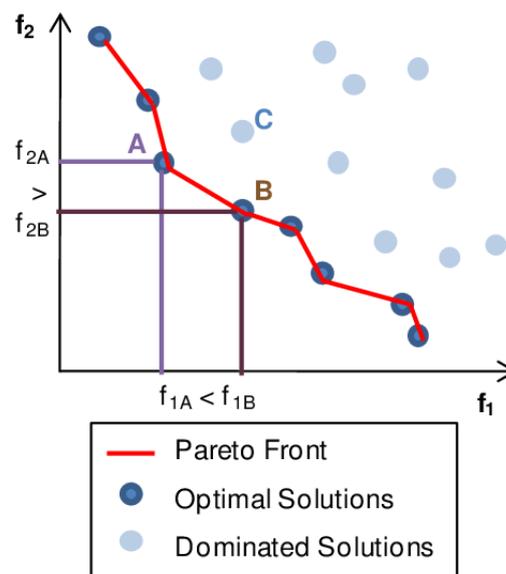


Figura 2.1 – Frente de Pareto (VILLA; LABAYRADE, 2011).

visualização, facilitando a compreensão das soluções que proporcionam os melhores resultados entre os diversos objetivos considerados em um problema multiobjetivo.

A utilização desses conceitos em otimização multiobjetivo é de suma importância para guiar a busca por soluções eficientes, pois o encontro de soluções Pareto-ótimas oferece um leque de possibilidades que permitem aos tomadores de decisão explorar e selecionar as alternativas mais adequadas, de acordo com suas preferências e restrições. Desse modo, a análise cuidadosa das soluções dominantes e da Fronteira de Pareto é um processo importante para a tomada de decisões e alcançar resultados satisfatórios em situações complexas e de múltiplos objetivos.

O problema central na otimização multiobjetivo, de acordo com Zheng e Wang (2021), reside na necessidade de encontrar uma única solução ótima de Pareto ou o conjunto completo de soluções ótimas de Pareto. A partir das soluções geradas, o tomador de decisão deve selecionar a mais promissora, considerando os requisitos específicos da aplicação. Para isso, o tomador de decisão tem algumas opções: ele pode configurar as preferências para cada objetivo, escolher a melhor solução de um conjunto de soluções ótimas de Pareto, ou ainda, tomar decisões ao longo do processo para, eventualmente, obter uma solução ótima de Pareto. Alternativamente, pode optar por não intervir no processo decisório, deixando o algoritmo responsável por produzir o conjunto de Pareto e selecionar a solução mais adequada.

Os algoritmos usados no contexto multiobjetivo podem ser classificados, de acordo com Zheng e Wang (2022), como métodos de *Scalarization* ou *Multi-objective Evolutionary Algorithm* (MOEA). A ideia central dos algoritmos de *Scalarization* é transformar vários objetivos em um único objetivo. Existem várias formas de realizar essa transformação, sendo uma das mais populares a utilização dos métodos de soma ponderada. Nesses métodos, atribuem-se pesos aos diferentes objetivos para representar a importância de cada um, e a soma ponderada é utilizada para representar um único objetivo a ser otimizado.

Por outro lado, segundo Coello, Veldhuizen e Lamont (2007), MOEA representa uma classe de algoritmos de otimização que aborda problemas multiobjetivo, simulando os princípios fundamentais do processo evolutivo em uma população de soluções. Essa abordagem é concretizada por meio dos operadores evolutivos, que englobam a atribuição de aptidão, seleção, cruzamento, mutação e elitismo. Os MOEAs frequentemente diferem em relação ao método de atribuição de aptidão, embora a maioria deles esteja agrupada na família denominada baseada em Pareto. Essa abordagem utiliza o conceito de dominância de Pareto como base para discriminar soluções e guiar sua busca.

A abordagem baseada em Pareto permite que os MOEAs identifiquem e mantenham um conjunto de soluções não dominadas, que representa as melhores soluções possíveis que atendem aos diferentes objetivos simultaneamente. Essas soluções na Fronteira de Pareto são consideradas igualmente boas, e a escolha final entre elas dependerá das preferências e restrições do tomador de decisão. Esse enfoque evolutivo permite explorar a diversidade das soluções e encontrar um conjunto de opções ótimas para resolver problemas complexos com múltiplos critérios.

No contexto deste projeto, os algoritmos classificados como MOEAs escolhidos são o *Non-dominated Sorting Genetic Algorithm II* (NSGA-II) (DEB et al., 2002) e o *Adaptive Genetic Evolution Multi-Objective Evolutionary Algorithm* (AGE-MOEA) (PANICHELLA, 2019). Cada um desses algoritmos possuem características distintas em seu funcionamento e abordagens específicas, proporcionando soluções eficazes para diferentes contextos.

O NSGA-II, adota uma abordagem evolutiva tradicional. Nesse método, a população evolui ao longo de gerações, empregando operadores genéticos, como seleção, cruzamento e mutação. Vale notar que as taxas desses operadores permanecem constantes ao longo da execução do algoritmo. Seu funcionamento, de acordo com Zheng e Wang (2021), se inicia gerando de forma aleatória uma população inicial de soluções. Cada solução é então avaliada em relação às funções objetivas do problema. Em seguida é realizada a classificação não dominada, onde as soluções são agrupadas em camadas de não dominação. As camadas de não dominação formam as frentes de Pareto, representando diferentes níveis de preferência. Os indivíduos que não são dominados por nenhum outro indivíduo, pertencem à primeira frente, enquanto os dominados pela primeira frente formam a segunda frente, e assim por diante.

A seleção de soluções para a próxima geração ocorre por meio de um processo de torneio entre as diferentes camadas de não dominação. Os operadores genéticos são aplicados aos indivíduos selecionados, gerando novas soluções. Segundo Deb (2011), primeiro com base nos valores de fitness atribuídos a cada solução, um processo de seleção é realizado para escolher quais soluções serão utilizadas para gerar a próxima geração. Então é realizado o *crossover*, que combina características de dois indivíduos diferentes, e a mutação, que introduz pequenas alterações nas características de um indivíduo. A próxima população é então formada pela combinação dos pais e dos descendentes resultantes do processo de cruzamento e mutação. Esses processos visam explorar diferentes regiões do espaço de busca, buscando melhorar as soluções

em termos dos objetivos definidos.

Uma característica do NSGA-II é sua capacidade de manter a diversidade na busca pela Fronteira de Pareto. A seleção das próximas gerações é realizada através da escolha de soluções de diferentes camadas de não dominação. As soluções da população são comparadas utilizando a distância de aglomeração (DEB et al., 2002). Garantindo que a busca pela Fronteira de Pareto seja diversificada.

O AGE-MOEA, segundo Panichella (2019) herda a estrutura geral do NSGA-II, incorporando modificações para aprimorar sua eficácia. Uma das mudanças principais no AGE-MOEA é a substituição da distância de aglomeração por uma pontuação de sobrevivência. Essa pontuação é projetada para integrar a distância entre vizinhos e a proximidade ao ponto ideal. Essa abordagem visa equilibrar a busca por soluções diversas e próximas, contribuindo para um conjunto mais representativo e eficiente de soluções ótimas.

Inicialmente, o algoritmo começa com uma população de soluções candidatas, avaliadas em termos de suas características de não dominação. Após a divisão da população em frentes não dominadas, as soluções na primeira frente, são utilizadas para a normalização do espaço objetivo e a estimativa da geometria da frente de Pareto.

O AGE-MOEA emprega uma heurística rápida com complexidade computacional  $O(M \times N)$ , onde  $M$  é o número de objetivos e  $N$  é o tamanho da população, para estimar a geometria da frente em cada geração, permitindo uma análise eficaz da estrutura das soluções não dominadas. Após a normalização, calcula-se a pontuação de sobrevivência que combina a diversidade, medida usando a distância entre os membros da população, e a proximidade das soluções nas frentes não dominadas, calculada como a distância entre cada membro da população e o ponto ideal.

O AGE-MOEA emprega uma estratégia de acasalamento por torneio binário para aumentar a pressão seletiva. Nesse processo, cada indivíduo é primeiramente avaliado com base em sua classificação e, posteriormente, é atribuída uma pontuação calculada que reflete tanto a proximidade quanto a dispersão em relação aos demais.

Diante dos conceitos apresentados, no contexto de Sistemas de Recomendação, uma das formas que se pode usar os algoritmos de otimização multiobjetivo é para balancear múltiplas métricas de avaliação (JANNACH; ABDOLLAHPOURI, 2023; ZHENG; WANG, 2021). Para avaliar a eficiência dos Sistemas de Recomendação, inúmeras métricas podem ser utilizadas, como será apresentado na Subseção 2.1.2. Entretanto, em muitos casos, esses objetivos podem ser considerados concorrentes, como demonstrado por Jannach e Abdollahpouri (2023).

Diversas propostas de estudos para equilibrar objetivos concorrentes em Sistemas de Recomendação podem ser encontrados na literatura. Dentre esses estudos, vale mencionar o trabalho de Zhang e Hurley (2008), no qual ele aborda a questão relacionada ao fato de que, ao fundamentar as listas de recomendação em métricas de similaridade, estas frequentemente apresentam itens excessivamente semelhantes entre si. Zhang e Hurley (2008) conceberam um

modelo que visa otimizar a diversidade sem comprometer a precisão das recomendações. Como resultado, houve um aprimoramento da utilidade das listas de recomendação para os usuários, fruto da implementação de estratégias de diversificação.

Outro exemplo, é o trabalho de [Ribeiro et al. \(2015\)](#), no qual o algoritmo de otimização utilizado foi o MOEA, com o objetivo de encontrar soluções ótimas que equilibrem as métricas de avaliação de precisão, diversidade e novidade. E nesse sentido exploraram o conceito de eficiência de Pareto, ao combinar algoritmos de recomendação de tal forma que a maximização de uma métrica não prejudicasse significativamente outra métrica considerando as relações de dominância entre os objetivos.

Por fim, o trabalho de [Fortes \(2022\)](#), cujo foco é explorar as preferências dos usuários em relação aos múltiplos objetivos otimizados de recomendação. A pesquisa aborda a questão de se é possível atender às diversas preferências individuais dos usuários em Sistemas de Recomendação. São utilizadas uma combinação de noções de Dominância de Pareto, Dominância Extrema e Testes de Significância Estatística. O objetivo é orientar a busca de soluções ótimas para produzir recomendações que se aproximem das preferências dos usuários. Para realizar a busca multiobjetivo, é utilizado o NSGA-II, descrito anteriormente.

### 2.1.2 Avaliação de Sistemas de Recomendação

Ao desenvolver um sistema de recomendação, torna-se crucial avaliar o desempenho dessas soluções considerando múltiplos aspectos. Essa avaliação visa confirmar a capacidade do sistema em sugerir itens pertinentes, alinhados com os objetivos específicos da aplicação. Além disso, a maioria dos pesquisadores que propõe novos algoritmos de recomendação realiza comparações com abordagens já estabelecidas. Para efetuar tais comparações, é comum empregar métricas de avaliação que permitem quantificar o desempenho relativo dos sistemas ([HERLOCKER et al., 2004](#); [GUNAWARDANA](#); [SHANI, 2009](#)).

Ao longo do tempo, o critério predominante para avaliar a eficácia dos Sistemas de Recomendação tem sido a acurácia, visando quantificar a precisão das sugestões alinhadas com as preferências de cada usuário ([VARGAS](#); [CASTELLS, 2011](#); [MCNEE](#); [RIEDL](#); [KONSTAN, 2006a](#)). Entretanto, atualmente reconhece-se que a satisfação do usuário nem sempre está diretamente ligada à alta acurácia do recomendador ([RASHID et al., 2002](#)). Há vários outros fatores importantes para os usuários que também necessitam ser considerados. Dessa forma, é de suma importância, ao avaliar um sistema de recomendação, atentar à satisfação do usuário com as sugestões oferecidas.

Em um sistema de recomendação de filmes, por exemplo, é comum os usuários desejarem sugestões alinhadas com seus gostos, mas que também tragam novidades. Nesse contexto, [McNee, Riedl e Konstan \(2006a\)](#) introduz o conceito de *similarity hole*, que ocorre quando as recomendações baseadas na similaridade entre itens são excessivamente semelhantes devido a

uma base de dados limitada. A precisão das previsões individuais, medida pela acurácia, não consegue abordar tais problemas.

Diante dessa perspectiva, a avaliação de Sistemas de Recomendação progrediu consideravelmente, incorporando conceitos igualmente significativos, como a novidade, que enfoca na capacidade do sistema de apresentar itens novos e pouco familiares ao usuário, a diversidade, que diz respeito às diferenças entre os itens de um conjunto, a serendipidade, que avalia a capacidade de surpreender positivamente o usuário apresentando itens inesperados, entre outras possibilidades. Introduzir elementos novos ou surpreendentes aprimora a experiência de recomendação, incentivando a exploração de novos interesses, enquanto a diversidade contribui para uma experiência mais variada.

Nesse trabalho, será abordada a tarefa de recomendação Top-N, e serão apresentadas as métricas a serem utilizadas nos experimentos. Como critério de novidade, será adotada a métrica *Expected Popularity Complement* (EPC) (VARGAS; CASTELLS, 2011), e como critério de acurácia, a métrica *Normalized Discounted Cumulative Gain* (NDCG) (JÄRVELIN; KEKÄLÄINEN, 2002).

A métrica EPC, segundo Vargas e Castells (2011), aborda a novidade considerando a popularidade inversa dos itens recomendados. Essa é uma métrica de avaliação de recomendação que tem em vista medir a novidade das recomendações fornecidas por um sistema de recomendação considerando tanto a relevância dos itens recomendados quanto a probabilidade de esses itens não terem sido previamente observados pelos usuários.

$$\text{EPC} = C \sum_{i_k \in R} \text{disc}(k) \cdot p(\text{rel} \mid i_k, u) \cdot (1 - p(\text{seen} \mid i_k)) \quad (2.2)$$

- $C$  é uma constante de normalização,
- $R$  representa o conjunto de itens recomendados,
- $i_k$  denota um item específico no conjunto de recomendações,
- $\text{disc}(k)$  é o desconto aplicado ao item na posição  $k$  da lista,
- $p(\text{rel} \mid i_k, u)$  é a probabilidade de relevância do item  $i_k$  para o usuário  $u$ ,
- $p(\text{seen} \mid i_k)$  é a probabilidade de o item  $i_k$  já ter sido visto pelo usuário.

O EPC é calculado somando a contribuição de cada item recomendado para o usuário, considerando sua relevância, sua posição na lista de recomendações e sua popularidade inversa. Quanto mais próximo de 1 for o resultado da métrica EPC, maior é a novidade das recomendações. Isso significa que o sistema está fazendo recomendações mais inovadoras ou menos populares entre os usuários. Essa abordagem proporciona uma avaliação da capacidade do sistema de

apresentar recomendações que fogem do convencional e introduzem itens menos conhecidos para os usuários.

Segundo Wang et al. (2013), a métrica NDCG mede a qualidade das listas de resultados retornadas pelos modelos, levando em consideração a relevância dos itens que os usuários indicaram como importantes ou relevantes. Em essência, o NDCG avalia não apenas se os itens recomendados são relevantes, mas também considera a ordem em que esses itens são apresentados na lista. Ele pondera as relevâncias dos itens pela posição em que aparecem na lista de recomendação, atribuindo maior peso aos itens mais relevantes que estão posicionados mais acima na lista.

$$DCG = \sum_{i=1}^k \frac{rel_i}{\log_2(i+1)} \quad (2.3)$$

- $rel_i$ : Relevância do item na posição  $i$  na lista de recomendações.
- $i$ : Posição do item na lista.
- $\log_2(i+1)$ : Fator de desconto logarítmico.

$$NDCG = \frac{DCG}{IDCG} \quad (2.4)$$

- $DCG$ : Mede a relevância acumulativa dos itens em uma lista de recomendações, dando mais peso aos itens relevantes em posições superiores, conforme definido pela Equação 2.3.
- $IDCG$ : Representa o valor máximo possível do DCG. Para calcular o IDCG, a lista de recomendações é ordenada por relevância e, em seguida, é aplicada a fórmula do DCG nessa lista ordenada.

A fórmula do NDCG normaliza o *Discounted Cumulative Gain* (DCG) pelo *Ideal Discounted Cumulative Gain* (IDCG), proporcionando uma pontuação normalizada que varia entre 0 e 1. Quanto mais próximo de 1, melhor é o desempenho do sistema de recomendação, indicando que os itens relevantes são apresentados no topo da lista.

Essa métrica é relevante para a tarefa de recomendação Top-N, pelo fato de considerar a ordenação dos itens na lista recomendada. Em cenários Top-N, a ordem dos itens desempenha um papel crucial, por refletir diretamente na experiência do usuário. Itens mais relevantes devem ser apresentados mais próximos do topo da lista para garantir uma recomendação eficaz.

## 2.2 Trabalhos Relacionados

Essa seção será iniciada pela Subseção 2.2.1, que se concentrará no **RecSysExp**, o principal foco desta pesquisa. Em seguida, nas Subseções 2.2.2 e 2.2.3, serão abordados os *frameworks* **RecMetrics** (BRUCKNER, 2017; SOUZA, 2014) e **Experimentor** (DIAS, 2019), respectivamente, que fazem parte do **RecSysExp**. Posteriormente, nas subseções seguintes, serão analisados outros trabalhos relacionados ao escopo deste estudo.

A revisão bibliográfica do *framework* **RecSysExp** permitirá a compreensão de suas funcionalidades, potencialidades e limitações, bem como a identificação de oportunidades de aprimoramento. Além disso, ao analisar outros trabalhos relacionados, será obtida uma visão mais abrangente dos *frameworks* e bibliotecas disponíveis, contribuindo para a evolução deste trabalho.

### 2.2.1 RecSysExp

Natali (2023) desenvolveu o *framework* **RecSysExp**, que foi projetado para facilitar a experimentação e a avaliação de Sistemas de Recomendação. Ele oferece uma série de recursos e técnicas para auxiliar os pesquisadores e desenvolvedores nesse processo. Facilitando a comparação e análise de diferentes abordagens de recomendação, permitindo a identificação das melhores técnicas e configurações para cada caso específico.

Este projeto foi inspirado na fusão de várias pesquisas relacionadas à área de Sistemas de Recomendação. A proposta é criar um *framework* abrangente que permita gerenciar, executar e reproduzir experimentos em Sistemas de Recomendação de forma mais eficiente. Entre os estudos que serão integrados para construir o **RecSysExp**, destaca-se o trabalho de SOUZA (2014), que desenvolveu o *framework* **RecMetrics** para calcular *meta-features* em Sistemas de Recomendação com base em Filtragem Colaborativa. BRUCKNER (2017) deu continuidade ao trabalho de SOUZA (2014), aprimorando o **RecMetrics** o deixando mais simples para implementar novas *features*. Além disso, incluiu métricas para sistemas Baseados em Conteúdo e realizou refatorações que resultou em uma melhora de *performance*.

Dias (2019) contribuiu, na parte de gestão e execução de experimentos, de forma prática e econômica propondo o *framework* **Experimentor**. Ele requer apenas a especificação das tarefas e suas respectivas entradas. O processo, desde a preparação até a execução e visualização dos resultados, é automatizado pelo próprio *framework*.

O **RecSysExp** disponibiliza uma grande variedade de funcionalidades que fornecem aos pesquisadores as ferramentas necessárias para realizar experimentos robustos e analisar os resultados. Com o **RecSysExp**, é possível medir o desempenho dos Sistemas de Recomendação, selecionar e manipular conjuntos de dados de maneira eficiente, validar os resultados por meio de testes estatísticos e interpretar os dados visualmente.

Além disso, o *framework* incentiva a reprodutibilidade, estabelecendo uma estrutura padronizada que permite a execução consistente e comparável de experimentos em diferentes estudos. Isso facilita a comparação de resultados e promove a transparência na pesquisa, permitindo que os pesquisadores analisem e interpretem os resultados de forma mais confiável.

O **RecSysExp** foi projetado para atender dois tipos de usuários: desenvolvedores e experimentadores, buscando ser de fácil utilização para ambos. O desenvolvedor é responsável por planejar, executar, analisar experimentos e desenvolver novos recursos. Enquanto isso, o experimentador conduzirá todas as etapas mencionadas anteriormente, exceto pelo desenvolvimento de novos recursos.

A arquitetura do processo de análise de dados possui três subdivisões essenciais: pré-processamento, modelagem e treinamento, e avaliação e visualização dos resultados. Cada uma dessas etapas desempenha um papel fundamental no sucesso da fase subsequente.

No estágio de pré-processamento, a base de dados é carregada, sendo necessário que esta esteja no formato de *dataframe* do Pandas, uma biblioteca Python dedicada à manipulação e análise de dados. A estrutura do *dataframe* deve conter os campos previstos por Natali (2023): *user*, com os identificadores e atributos dos usuários, *item*, com os identificadores e atributos dos itens, e *rating*, contendo as avaliações atribuídas pelos usuários.

Por exemplo, consideremos um conjunto de dados que represente produtos de uma loja de *e-commerce*. Nesse cenário, as avaliações dos produtos são representadas por uma matriz de avaliações realizadas pelos usuários, enquanto as informações dos usuários são armazenadas em uma matriz que contém características como idade e sexo. Da mesma forma, os itens também são representados em uma matriz que inclui informações como descrição, valor e nicho de cada produto.

Após o carregamento dos dados, o *framework* oferece diversas operações que possibilitam a manipulação e adequação dos mesmos para a fase de modelagem e treinamento. A estrutura do projeto inclui arquivos que implementam classes capazes de realizar tarefas como a divisão dos dados em conjuntos de treino e teste, normalização, discretização, e outras estratégias relevantes. Essas operações são essenciais para garantir que os dados estejam prontos para serem utilizados de forma eficaz no processo de análise e desenvolvimento do modelo.

Na etapa de modelagem e treinamento, encontram-se todos os algoritmos de recomendação desenvolvidos, constituindo a base para todo o processo de treinamento, predição e recomendação. Nessa parte, a arquitetura é feita de forma genérica. Por meio das interfaces e classes abstratas previamente definidas, torna-se possível a adição de métodos específicos de maneira modular. Esta estrutura genérica foi delineada, de modo a capacitar o *framework* a suportar algoritmos que englobem filtragem colaborativa, filtragem baseada em conteúdo e a filtragem híbrida.

Também é realizado o cálculo das *meta-features*, utilizando a ferramenta **RecMetrics**

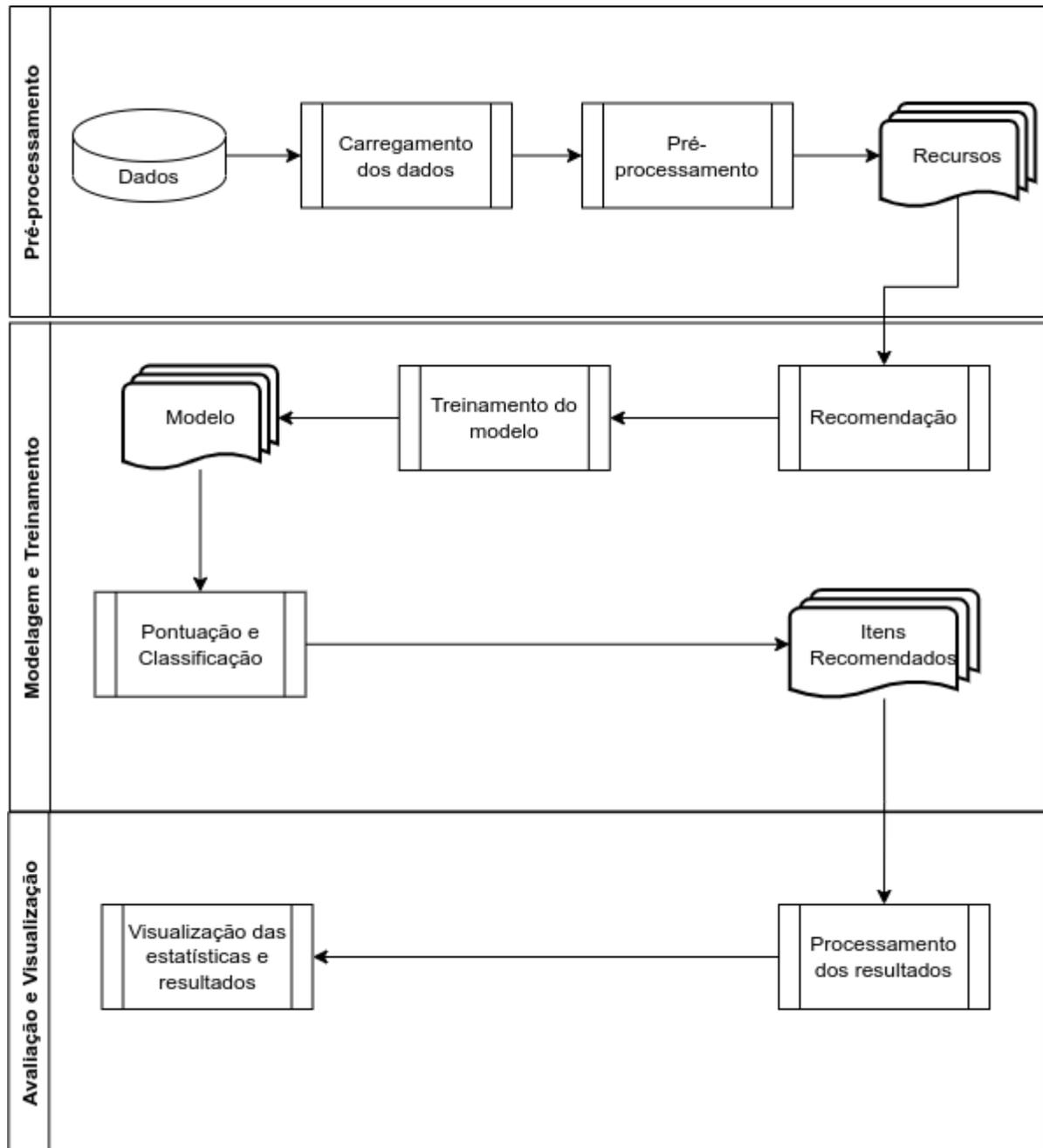


Figura 2.2 – Arquitetura Geral RecSysExp (NATALI, 2023).

(BRUCKNER, 2017; SOUZA, 2014). Algumas configurações para a execução são definidas para cada processo a ser realizado. É necessário especificar informações como o tipo de processamento, baseado em conteúdo ou filtragem colaborativa, e as *meta-features* a serem consideradas. Após os cálculos, o **RecMetrics** gera uma saída composta por diversos arquivos de texto, cada um separado pelo tipo de *meta-feature* e com o nome conforme especificado no arquivo de configuração.

Por fim, adentramos à etapa de avaliação e visualização. Uma vez obtidos os resultados provenientes do treinamento dos modelos de recomendação, estes são submetidos a um conjunto diversificado de métodos de avaliação. Nessa fase, o framework mantém uma arquitetura de caráter genérico, facilitando a inclusão de novas métricas à medida que necessário. Dentre as

métricas já incorporadas ao framework, destacam-se o *Root Mean Squared Error (RMSE)*, *Mean Absolute Error (MAE)*, *Normalized Discounted Cumulative Gain (NDCG)*.

Na etapa subsequente do processo, os resultados provenientes das avaliações são analisados pelo módulo responsável por conduzir os cálculos estatísticos. Esse módulo abrange métodos como intervalo de confiança e análise de variância.

Após obtermos os resultados das métricas e realizar os cálculos estatísticos, avançamos para a etapa de visualização das estatísticas e resultados. Essa visualização é fundamental para uma compreensão mais clara dos dados analisados, tornando-os mais acessíveis e facilitando a identificação de padrões e tendências relevantes. O *framework* tem uma estrutura que permite gerar tabelas, gráficos de pizza ou de barras, com opções interativas ou estáticas. Essa variedade de recursos permite apresentar os resultados de forma mais dinâmica e atrativa, contribuindo para uma melhor comunicação e interpretação dos achados obtidos na análise.

Após apresentar a arquitetura do *framework*, é relevante enfatizar que o **RecSysExp** possui algumas limitações. Uma delas é o seu escopo limitado de aplicações. O *framework* foi principalmente projetado para Sistemas de Recomendação baseados em filtragem colaborativa e baseada em conteúdo. Outros tipos de recomendação, como os multiobjetivo, não são suportados pelo **RecSysExp**. Ainda nesse sentido, o *framework* se limita também por não ter métricas que avaliem fatores como novidade, diversidade, ou serendipidade nas recomendações. Portanto, é essencial avaliar cuidadosamente se o *framework* atende às necessidades específicas do projeto de pesquisa em questão.

Outra limitação diz respeito ao uso do *framework* e às dependências necessárias para sua implementação. Durante a utilização do **RecSysExp** com alunos da disciplina Sistemas de Recomendação (BCC409) do Departamento de Computação da Universidade Federal de Ouro Preto, Natali (2023) identificou desafios relacionados à instalação das dependências, uma vez que algumas bibliotecas requeriam versões específicas de outras bibliotecas.

Apesar dessas limitações, o **RecSysExp** continua sendo uma ferramenta valiosa para a condução de experimentos em Sistemas de Recomendação, fornecendo recursos que contribuem para a qualidade e confiabilidade da pesquisa nessa área. O trabalho de Natali (2023) representa uma contribuição significativa para o campo dos Sistemas de Recomendação, ao fornecer um *framework* abrangente e flexível que simplifica o processo de implementação e validação desses sistemas. Com o **RecSysExp**, os pesquisadores têm à disposição uma ferramenta prática para conduzir experimentos e aprimorar continuamente os Sistemas de Recomendação.

## 2.2.2 RecMetrics

O *framework* **RecMetrics** teve origem a partir do trabalho de SOUZA (2014), visando calcular *meta-features* em Sistemas de Recomendação baseados em Filtragem Colaborativa. A implementação desse *framework* incorporou técnicas de programação paralela para otimizar

o processamento de grandes volumes de dados. No entanto, a versão inicial do *framework* apresentava algumas limitações, restringindo-se à Filtragem Colaborativa e enfrentando desafios na inclusão de novas métricas.

Para superar essas limitações, BRUCKNER (2017) deu continuidade ao desenvolvimento do *framework*, promovendo uma expansão abrangente. Isso incluiu a adição de métricas para sistemas baseados em conteúdo, bem como uma refatoração do código para facilitar a incorporação de novas métricas.

A ampliação do *framework* foi fundamentada na arquitetura original proposta por SOUZA (2014). Para configurar e executar as métricas de maneira adequada, é necessário definir um arquivo *Extensible Markup Language* (XML) contendo detalhes sobre o tipo de processamento, seja baseado em conteúdo ou filtragem colaborativa, e as *meta-features* a serem consideradas. Com as configurações definidas, o *framework* **RecMetrics** consegue gerar os resultados dos cálculos das métricas.

O cálculo das *meta-features* varia dependendo da abordagem escolhida. No contexto da filtragem colaborativa, as *meta-features* são obtidas a partir das avaliações numéricas fornecidas pelos usuários para os itens. O módulo encarregado desse cálculo é construído de forma genérica, permitindo a criação de subclasses que incorporam a lógica específica de cada *meta-feature*. Por exemplo, uma das *meta-features* utilizadas, chamada *Proportion of Ratings*, está definida na estrutura do sistema e indica a proporção das avaliações atribuídas a um item ou usuário em relação ao total.

No caso das métricas baseadas em conteúdo, o cálculo é realizado com base nos dados descritivos dos itens. Isso requer a extração de informações essenciais, como frequência de termos e similaridade entre itens. Além disso, é incorporada uma etapa adicional para a indexação dos dados. A arquitetura nesse contexto também é modelada de maneira genérica, possibilitando a incorporação das *meta-features* específicas do contexto. Um exemplo presente no *framework* é a Entropia de Shannon, responsável por calcular a coesão do conteúdo presente em um documento.

Após o cálculo das *meta-features*, tanto na filtragem colaborativa quanto na baseada em conteúdo, a saída consiste em vários arquivos de texto separados por tipo de *meta-feature*, conforme especificado no arquivo de configuração.

### 2.2.3 Xperimentor

O *framework* **Xperimentor** nasceu com o propósito de gerenciar as tarefas de execução de experimentos computacionais. Desenvolvido por Dias (2019), o *framework* **Xperimentor** foi concebido para atender à necessidade de cientistas, pesquisadores e profissionais que desejam conduzir seus experimentos computacionais de forma mais fluida. O **Xperimentor** automatiza cada etapa do processo de gestão do fluxo do experimento, organizando as tarefas desde a configuração inicial até a apresentação dos resultados finais. Isso permite que os usuários se

concentrarem no desenvolvimento e análise dos próprios experimentos.

Uma característica notável do *framework* **Xperimentor** é sua configuração simplificada. Utilizando o **Xperimentor** para gerenciar e executar experimentos, é suficiente fornecer as especificações das tarefas e suas relações. O próprio *framework* se encarrega de criar um grafo de dependências baseado nessas especificações, determinando de forma automática a ordem correta de execução das tarefas. O **Xperimentor** utiliza esses detalhes para construir um fluxo de trabalho otimizado, assumindo tanto a execução das tarefas quanto a apresentação dos resultados alcançados.

A arquitetura do *framework* **Xperimentor** é visualmente elucidada na Figura 2.3. O *framework* é composto por dois módulos. O *Front-end*, denominado **Xperimentor**, é responsável por gerir a execução dos experimentos e disponibilizar uma interface ao experimentador. Através de uma página, os usuários podem interagir com um editor de código embutido e um painel de visualização, o que lhes permite monitorar o progresso do experimento em tempo real. O *Back-end*, ou **Task Executor**, é implantado em um *cluster Kubernetes*, onde as tarefas do experimento são executadas. Quando uma tarefa é programada para execução, ela é redirecionada para uma máquina no *cluster*, onde é efetivamente processada.

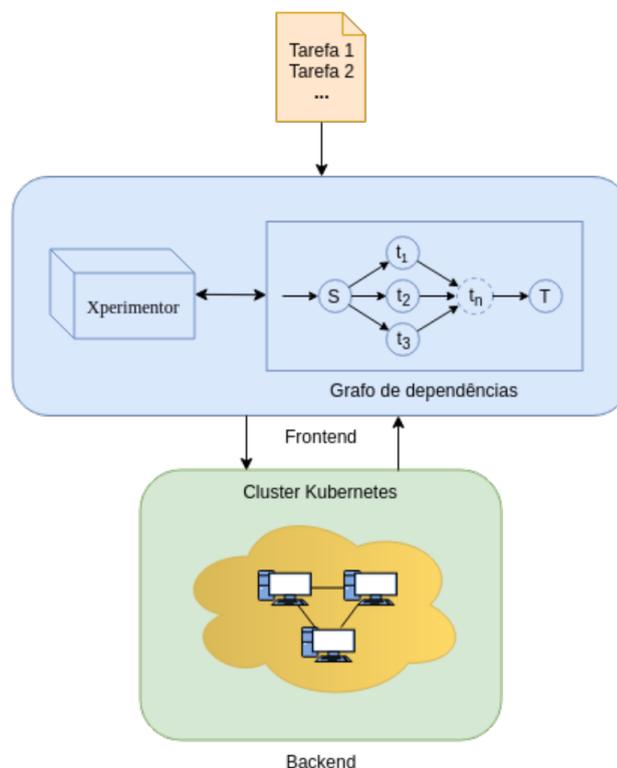


Figura 2.3 – Arquitetura Xperimentor (DIAS, 2019).

Com o *framework* **Xperimentor**, a complexidade da gestão de experimentos computacionais é reduzida significativamente, ao automatizar todo o fluxo de trabalho, o **Xperimentor** se estabelece como uma ferramenta de grande utilidade no arsenal de pesquisadores.

## 2.2.4 RankSys

O *framework* **RankSys**, proposto por Vargas e Castells (2011), é utilizado para a avaliação e aprimoramento dos Sistemas de Recomendação, com um foco específico na novidade e diversidade das recomendações. Essa estrutura fornece uma variedade de modelos e métricas que permitem a comparação e a análise dos resultados obtidos de vários algoritmos de recomendação.

O **RankSys** emprega dois tipos de métricas para avaliar a eficácia dos algoritmos de recomendação, a saber, métricas orientadas ao usuário e métricas orientadas ao sistema. As métricas orientadas ao usuário medem até que ponto uma recomendação específica atende às necessidades do usuário que a recebe, enquanto as métricas orientadas ao sistema medem a eficácia geral de um conjunto de recomendações emitidas para uma comunidade de usuários. A estrutura inclui uma variedade de métricas comumente usadas em tarefas de classificação, como *precision*, *recall*, *F-score*, *average precision*, *reciprocal rank*, *k-call* e *NDCG*, para avaliar a qualidade das recomendações. Além disso, o *RankSys* fornece várias métricas e abordagens para avaliar a novidade e a diversidade das recomendações.

Para avaliar a novidade dos itens recomendados, o **RankSys** utiliza métricas como *popularity complement (PC)*, *free discovery (FD)* e *profile distance (PD)*. Além disso, para avaliar a diversidade, o *framework* aborda métricas de diversidade agregada, que calcula a diversidade com base no número total de itens recomendados para uma comunidade de usuários. As métricas implementadas incluem a *Entropy*, o *Gini Index* e o *Gini-Simpson Index*.

Em resumo, o **RankSys** é uma estrutura abrangente que oferece um conjunto diversificado de métricas e modelos para avaliar e aprimorar os Sistemas de Recomendação, com um foco particular na novidade e diversidade das recomendações. Sua versatilidade e variedade de abordagens o tornam uma ferramenta valiosa para pesquisadores e profissionais que desejam explorar e aprimorar a qualidade e a diversidade dos Sistemas de Recomendação.

## 2.2.5 Elliot

O *framework* **Elliot**, proposto por Anelli et al. (2021), fornece um ambiente estruturado para realizar o processo de experimentação em Sistemas de Recomendação. Sua arquitetura é composta por uma série de componentes fundamentais que abrangem etapas cruciais do processo de criação de Sistemas de Recomendação. Esses componentes incluem o pré-processamento dos dados, a implementação de métodos de recomendação, a definição de métricas de avaliação do modelo e a realização de testes estatísticos.

Por meio de um arquivo de configuração, os desenvolvedores podem definir parâmetros específicos para o experimento, garantindo a reprodutibilidade dos resultados obtidos. Essa abordagem modular e flexível permite que os desenvolvedores combinem e ajustem diferentes componentes para atender às necessidades específicas de seus projetos. O *framework* **Elliot** suporta uma ampla variedade de algoritmos de recomendação. Isso proporciona aos desenvolve-

dores uma grande diversidade de opções para criar sistemas altamente eficientes e personalizados de recomendação.

O *framework* **Elliot** se assemelha com a proposta apresentada neste estudo. Tanto o **Elliot** quanto o **RecSysExp** compartilham semelhanças na utilização de um arquivo de configuração para gerenciar a execução do experimento e ambos possuem uma estrutura que permite a execução e reprodução de todo o *pipeline* experimental.

No entanto, existem diferenças significativas entre essas duas abordagens. Enquanto ambas têm em comum o objetivo de permitir a execução de todo o processo experimental, o **RecSysExp** busca um propósito adicional. Seu foco é estabelecer uma estrutura genérica que segregue por categorias, como exemplificado pelo módulo de recomendação com estruturas genéricas para diferentes tipos de filtragem. A partir dessa estrutura, o **RecSysExp** tem em vista viabilizar a inclusão de métodos mais específicos, aproveitando a riqueza de recursos disponíveis.

Em relação às abordagens de recomendação, vale notar que o **Elliot** carece de métodos embutidos com enfoque em filtragem multiobjetivo, sendo um dos pontos focais deste estudo. Adicionalmente, o **RecSysExp** se distingue ao oferecer um módulo de visualização que interpreta os resultados produzidos, criando representações visuais correspondentes. Isso melhora a compreensão e a análise dos resultados obtidos nos experimentos de forma mais detalhada.

Ademais, a estrutura proposta se destaca por ter a possibilidade da utilização de *meta-features*, em que características adicionais dos conjuntos de dados de entrada são extraídas para complementá-los e melhorar as recomendações. Nesse sentido, ter a possibilidade de usar *meta-features* no *framework* **RecSysExp** é um valioso diferencial.

### 2.2.6 DaysiRec 2.0

O *framework* **DaisyRec** é uma ferramenta relevante no campo de Sistemas de Recomendação. Ele foi projetado segundo Sun et al. (2022) visando fornecer uma estrutura padronizada para avaliar e comparar algoritmos de recomendação de maneira rigorosa e eficiente. O foco principal do *framework* é possibilitar uma avaliação de Sistemas de Recomendação confiável e reproduzível, com isso o **DaisyRec** se torna uma ferramenta valiosa.

A flexibilidade do **DaisyRec** é um recurso proeminente que permite acomodar uma variedade de algoritmos de recomendação, desde os mais básicos, baseados em filtragem colaborativa, até os mais avançados, como técnicas de aprendizado profundo. Essa versatilidade permite que os pesquisadores examinem diferentes estratégias e abordagens, adaptando assim a estrutura a problemas específicos e desafiadores.

A arquitetura do *framework* consiste em três módulos: *Loader*, *Recommender* e *Evaluator*. O *Loader* carrega e pré-processa os dados, dividindo-os em conjuntos de treinamento e teste. O *Recommender* contém os métodos de recomendação, enquanto o *Evaluator* é responsável pelas métricas de avaliação. Todos os módulos são encapsulados de forma que novos métodos possam

ser incluídos, tornando o *framework* extensível.

A reprodutibilidade de resultados é um dos pilares fundamentais do **DaisyRec**. Os pesquisadores podem facilmente reproduzir os experimentos realizados no *framework*, garantindo que os resultados sejam confiáveis e que outros possam verificar e validar as descobertas. Essa transparência é essencial para a construção de um conhecimento sólido na área de Sistemas de Recomendação.

Apesar de todas as vantagens, o **DaisyRec** apresenta algumas limitações. Embora o objetivo principal do **DaisyRec** seja promover uma avaliação rigorosa, com foco em métricas bem estabelecidas, ele pode não ser tão abrangente quando se trata de considerar aspectos de diversidade, novidade, serendipidade, dentre outras possibilidades que poderiam ser exploradas ao avaliar as recomendações.

A diversidade e a novidade são fatores críticos em Sistemas de Recomendação, pois possuem um papel fundamental na satisfação do usuário e na experiência geral do usuário com o sistema. A diversidade garante que o sistema sugira não apenas itens populares e óbvios, mas também recomendações que atendam aos diferentes interesses e preferências do usuário. A novidade, por outro lado, é importante para apresentar aos usuários itens inesperados e surpreendentes, aumentando seu engajamento e a descoberta de novos conteúdos ou produtos.

As métricas de avaliação tradicionais, como a acurácia, são cruciais para medir a qualidade geral das recomendações. No entanto, elas tendem a se concentrar apenas em quão bem o sistema pode prever as preferências do usuário com base em interações passadas, sem considerar adequadamente a diversidade e a novidade das recomendações. Como pontuado em (MCNEE; RIEDL; KONSTAN, 2006b) as recomendações mais precisas conforme as métricas padrão às vezes não são as recomendações mais úteis para os usuários.

Em suma, o *framework* **DaisyRec** representa uma contribuição significativa para a área de Sistemas de Recomendação, oferecendo uma infraestrutura padronizada e confiável para avaliar e comparar algoritmos. Sua flexibilidade, métricas de avaliação e ênfase na reprodutibilidade fazem dele uma ferramenta importante para impulsionar a pesquisa e o desenvolvimento de Sistemas de Recomendação mais eficientes e personalizados. No entanto, é importante estar ciente das limitações do *framework* ao interpretar os resultados e considerar a complementação com outras abordagens quando necessário.

### 2.2.7 Pymoo: Multi-objective Optimization in Python

O **Pymoo** é uma poderosa biblioteca Python que auxilia na resolução de problemas complexos de otimização multiobjetivo, incluindo aqueles relacionados a Sistemas de Recomendação. Através do **Pymoo**, os usuários têm acesso a uma ampla variedade de algoritmos de otimização multiobjetivo, permitindo que escolham a abordagem mais adequada para suas necessidades específicas.

Uma das principais vantagens do **Pymoo** é a sua facilidade de uso, uma vez que foi desenvolvido visando tornar a experiência do usuário amigável e acessível. A biblioteca também se destaca por ser extensível e modular, possibilitando implementações personalizadas conforme as demandas individuais dos usuários. Além disso, o **Pymoo** oferece uma documentação abrangente, com guias e explicações detalhadas, facilitando assim o processo de utilização (BLANK; DEB, 2020).

Para abordar desafios multiobjetivo em Sistemas de Recomendação, o **Pymoo** oferece a implementação de algoritmos genéticos multiobjetivo, incluindo o *Non-dominated Sorting Genetic Algorithm II* (**NSGA-II**) (DEB et al., 2002). Além disso, o **pymoo** conta com algoritmos fundamentados na decomposição multiobjetivo, como o *Multi-Objective Evolutionary Algorithm based on Decomposition* (**MOEA/D**) (ZHANG; LI, 2007). A biblioteca também dispõe de métodos para facilitar a tomada de decisão em relação às soluções na frente de Pareto, além de oferecer ferramentas para realizar análises de convergência.

No contexto do presente estudo, tem-se a intenção de utilizar os algoritmos disponíveis no **Pymoo** para integrar abordagens de recomendação multiobjetivo ao **RecSysExp**. Essa integração representa uma contribuição significativa para um dos objetivos fundamentais deste trabalho, o qual é incluir métodos que possam ser utilizados em Sistemas de Recomendação multiobjetivo.

## 3 Desenvolvimento

Neste capítulo, será detalhada a metodologia para alcançar os objetivos estabelecidos no presente trabalho. Em primeiro lugar, na Seção 3.1, é abordada a incorporação da métrica de novidade, aprimorando a avaliação das recomendações geradas pelo *framework* e viabilizando a aplicação da abordagem de recomendação multiobjetivo. Em seguida, na Seção 3.2, é detalhada a inclusão de métodos multiobjetivo no **RecSysExp**.

### 3.1 Inclusão da métrica de novidade

Na arquitetura relacionada às métricas de avaliação no *framework* **RecSysExp**, existe a interface principal denominada *Metric*. Essa interface desempenha um papel fundamental ao definir dois métodos essenciais: o método *evaluate*, que é utilizado para avaliar uma métrica com base nas previsões, e o método *checkMissing*, empregado para identificar valores ausentes nos dados. A função desta interface é atuar como um alicerce comum, servindo como base para todas as métricas incorporadas no sistema.

Em um nível intermediário, encontra-se a classe abstrata *AbstractMetric*, que herda os atributos definidos na interface *Metric*. Prosseguindo na hierarquia, deparamo-nos com a classe abstrata *PredictionMetric*, que serve como base para as classes concretas que efetivamente definem as métricas. Como exemplos dessas métricas, destacam-se RMSE e MAE.

Outra classe abstrata que compõe o **RecSysExp** é a *RankingMetric*, dedicada à avaliação de métricas relacionadas ao ranqueamento. Essas métricas são fundamentais para medir a qualidade das recomendações em termos de ordem de relevância. Um exemplo notável é o *Normalized Discounted Cumulative Gain* (NDCG), que considera tanto a relevância quanto a posição dos itens recomendados.

A partir das classes abstratas mencionadas, é possível derivar classes concretas que representam métricas específicas. Essas classes concretas são responsáveis por implementar o método *evaluate*, adaptando-o às características distintas de cada métrica. No escopo deste trabalho, foi incorporado como métrica de novidade o *Expected Popularity Complement* (**EPC**), apresentado por Vargas e Castells (2011). Para incluir essa métrica, foi criada a classe concreta *EPC*, derivada da classe abstrata *RankingMetric*. A arquitetura desse módulo de métricas pode ser visualizada na Figura 3.1.

O EPC, em sua forma original conforme apresentado na Equação 2.2, avalia a novidade em relação à popularidade esperada, levando em consideração a ordem dos itens recomendados. Essa métrica analisa como itens menos populares são posicionados em lugares mais altos, incentivando a apresentação de recomendações inovadoras aos usuários.

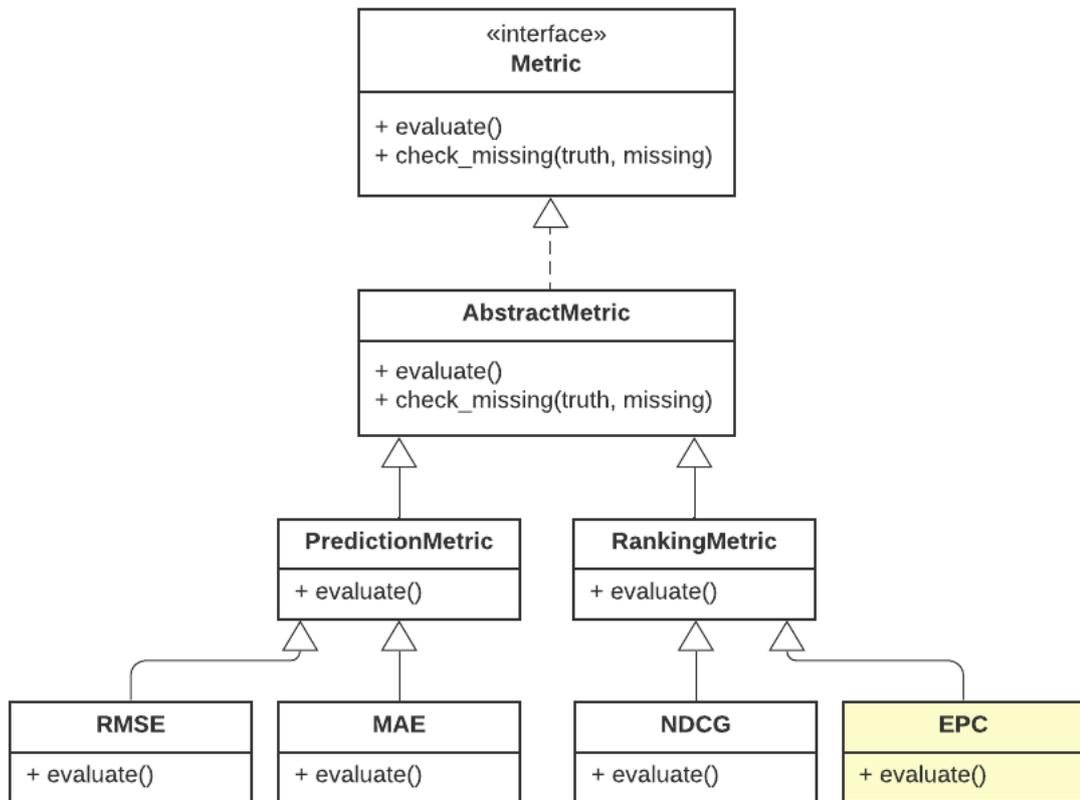


Figura 3.1 – Representação da arquitetura do módulo de métricas.

Entretanto, a classe EPC no *framework* oferece a flexibilidade adicional ao usuário que permite ajustar o cálculo da métrica, ignorando a relevância e o fator de desconto em relação a ordem se desejado. Ao instanciar a classe, é possível passar o método desejado para o cálculo, e se nenhum método específico for fornecido, a implementação padrão na classe prevalecerá. Essa flexibilidade oferece uma abordagem versátil para avaliação, adaptando-se às necessidades específicas do usuário ou do cenário de recomendação em questão.

A implementação do Algoritmo 1 segue a fórmula do EPC, conforme descrita na Equação 2.2. O processo se inicia com a criação de um dicionário de frequência dos itens, mapeando a quantidade de avaliações recebidas por cada item. Em seguida, esse dicionário é utilizado para gerar outro que representa a pontuação de novidade, sendo o inverso da popularidade de cada item, conforme calculado na Equação 3.1:

$$\text{novelty\_score} = 1 - \frac{\text{frequency}}{\text{num\_users}} \quad (3.1)$$

Em seguida, o algoritmo itera sobre cada item de *Predictions*, para obter sua relevância, desconto e pontuação de novidade. A determinação da relevância de um item para o usuário envolve a verificação se o item foi previamente visualizado pelo usuário e avaliado positivamente,

**Algorithm 1** EPC Metric**Input:** Predictions, Ratings**Output:** Valor do EPC calculado: *epc\_score*


---

```

1 norm ← 0
2 item_frequency_dict ← generateItemFrequencyDict(Ratings)
3 item_novelty_dict ← generateItemNoveltyDict(item_frequency_dict)
4 foreach rank, item in Predictions do
5   | relevance ← getRelevance(item, Ratings, threshold)
6   | discount ← getDiscount(rank)
7   | novelty_score ← getNoveltyValue(item, item_novelty_dict)
8   | epc_score ← epc_score + (discount * relevance * novelty_score)
9   | norm ← norm + discount
10 end
11 if norm > 0 then
12 | epc_score /= norm
13 end
14 return epc_score

```

---

isto é, se o *rating* ultrapassa o valor estipulado em *threshold*. Esse critério leva em consideração as interações prévias do usuário com as recomendações. Caso o *rating* de um item esteja acima do limite estabelecido pelo *threshold*, o sistema de recomendação o considerará como relevante para o usuário. Em contrapartida, se o *rating* estiver aquém do *threshold*, o item não será considerado relevante. Essa abordagem visa incorporar a história de interações do usuário com os itens recomendados.

Para calcular o fator de desconto, consideramos a posição do item na lista de recomendações, aplicando descontos menores a itens em posições mais altas. Esta abordagem reflete a intuição de que itens no topo da lista têm um impacto mais significativo na experiência do usuário, conforme expresso pela Equação 3.2:

$$\text{reciprocal\_discount} = \frac{1}{\text{rank} + 1} \quad (3.2)$$

A pontuação de novidade é obtida recuperando seu valor previamente calculado e armazenado no dicionário de novidade. Com essas informações, o cálculo do *epc\_score* é realizado multiplicando o desconto, a relevância e a pontuação de novidade. O resultado é acumulado para cada item e, ao final, é feita a normalização. Essa abordagem favorece a recomendação de itens menos populares, aumentando a probabilidade de oferecer novidades aos usuários.

## 3.2 Inclusão de métodos multiobjetivo

O *framework* **RecSysExp**, possui uma arquitetura composta por uma série de interfaces e classes abstratas que fornecem uma base sólida para a criação de algoritmos de recomendação

personalizados. Ao estender essas interfaces e classes definidas, é possível implementar algoritmos de recomendação de maneira estruturada e coesa.

Partindo da classe abstrata *AbstractRecommender*, surgem outras classes abstratas que representam estratégias de recomendação específicas. São elas: *AbstractCollaborativeRecommender*, *AbstractContentBasedRecommender* e *AbstractHybridRecommender*. Cada uma dessas classes abstratas fornece uma base para a implementação de algoritmos que se enquadram em suas respectivas abordagens. A visualização dessa arquitetura pode ser vista na Figura 3.2.

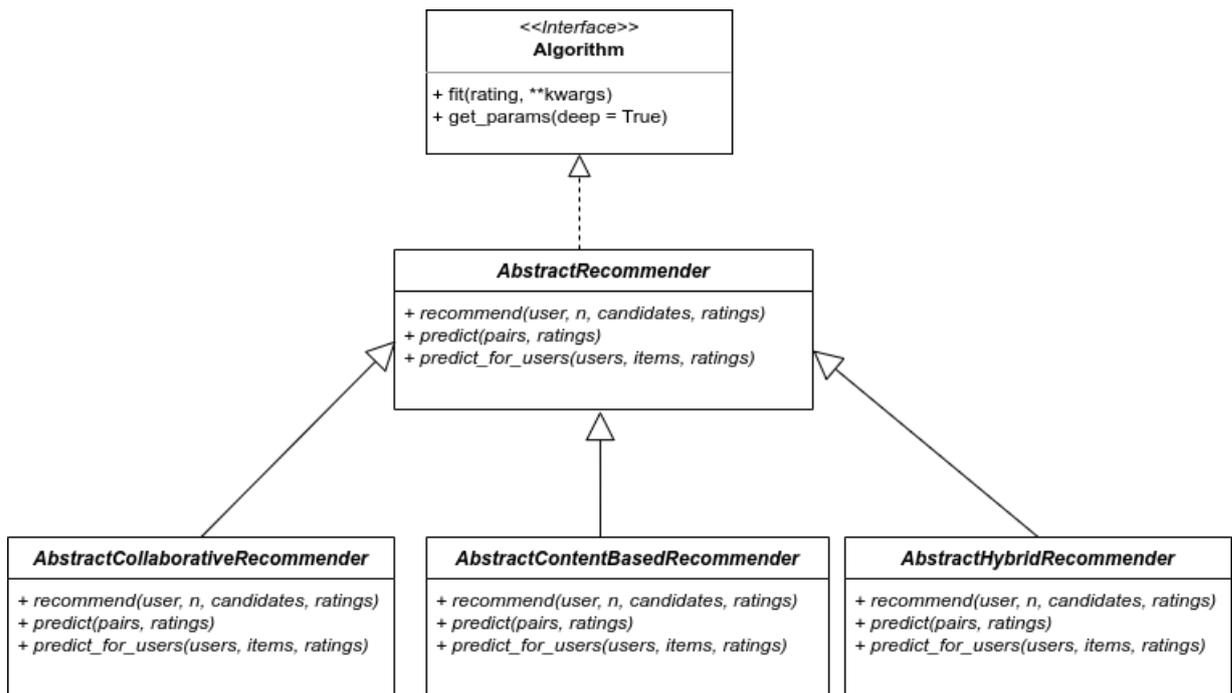


Figura 3.2 – Representação da arquitetura do módulo de recomendação (NATALI, 2023).

Para expandir ainda mais essa estrutura, foi incluída uma nova classe abstrata chamada *AbstractMultiObjectiveRecommender*, como pode ser visto na Figura 3.3. Através dessa classe, foram criadas classes concretas que incorporaram algoritmos de recomendação que operam com objetivos múltiplos. É importante destacar que, devido à natureza única de cada algoritmo, essas classes podem ter características específicas para melhor se adaptar aos requisitos individuais de cada algoritmo.

No contexto da otimização multiobjetivo aplicada em Sistemas de Recomendação, o processo se desenrola em duas etapas distintas. Inicialmente, há a fase de otimização, seguida pelo estágio de predições. Primeiramente, realiza-se a busca pelo conjunto de soluções não dominadas. Em seguida, ocorre a etapa de tomada de decisão para escolher uma solução, culminando no processo de predição, no qual são geradas listas de recomendações para os usuários com base na solução selecionada.

Foram instanciadas classes concretas da abstração *AbstractMultiObjectiveRecommender* no *framework* **RecSysExp**, em que foram incluídos os métodos de otimização multiobjetivo

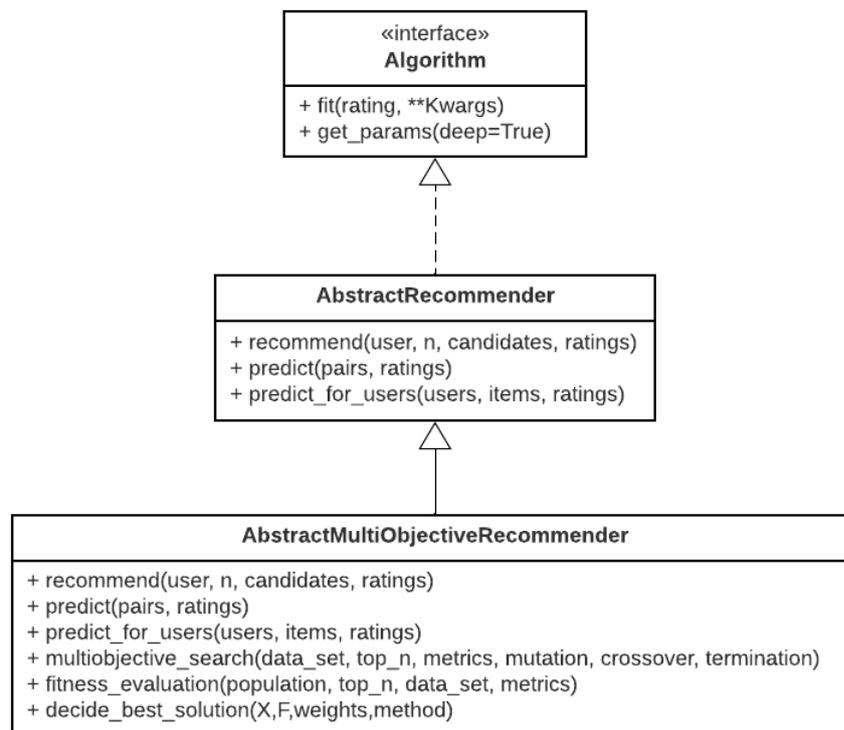


Figura 3.3 – Arquitetura do módulo incluindo suporte para métodos multiobjetivo.

*Non-dominated Sorting Genetic Algorithm II* (NSGA-II) (DEB et al., 2002) e o *Adaptive Genetic Evolution Multi-Objective Evolutionary Algorithm* (AGE-MOEA) (PANICHELLA, 2019). Todos compartilham uma estrutura que adere a uma mesma lógica de funcionamento. Para incluir esses métodos, foi utilizado a biblioteca Pymoo, uma ferramenta robusta que oferece métodos para resolver problemas de otimização multiobjetivo.

### 3.2.1 Otimização Multiobjetivo

A otimização multiobjetivo compreende duas tarefas cruciais: a busca por um conjunto de soluções de Pareto e a tomada de decisão entre as soluções não dominadas resultantes. Para integrar métodos multiobjetivos ao *framework*, concentramos nossa atenção nessas duas áreas. Nas subseções a seguir, será abordado a definição do problema, a fase de busca e a etapa de tomada de decisão.

#### 3.2.1.1 Definição do problema

Para a definição do problema de otimização, é essencial considerar várias características do modelo. Estas incluem o número de objetivos, a quantidade de variáveis em uma solução, os limites mínimo e máximo que especificam o intervalo de valores para as soluções, e as restrições, se existirem.

Neste trabalho, a otimização multiobjetivo é aplicada ao contexto de sistemas de recomendação. Os objetivos a serem otimizados são as métricas que avaliarão as recomendações, sendo o número de objetivos equivalente à quantidade de métricas a serem aplicadas. Optaremos por duas métricas que avaliam a acurácia e novidade das recomendações, resultando em dois objetivos a serem otimizados.

A abordagem escolhida para este trabalho é a modelagem de hibridização ponderada. Cada indivíduo ou solução é representado por um conjunto de pesos, cujo tamanho é determinado pela quantidade de *features* da base de dados utilizada. Cada peso varia no intervalo de 0 a 1, refletindo a importância relativa das *features* na composição da solução.

Dessa forma, a definição do problema de otimização é guiada pela necessidade de encontrar conjuntos de pesos que maximizem simultaneamente a acurácia e a novidade nas recomendações do sistema. Este enfoque permite uma abordagem equilibrada e eficiente na busca por soluções que atendam aos requisitos específicos do sistema de recomendação.

### 3.2.1.2 Etapa de busca

O Algoritmo 2 segue a estrutura requisitada pelo Pymoo para realizar o processo de busca do conjunto de soluções não dominadas. Ao iniciar a otimização com a função *minimize*, uma funcionalidade fornecida pelo Pymoo para otimização multiobjetivo (linha 4), são necessários os parâmetros *problem*, *algorithm* e *termination*.

O parâmetro *problem* (linha 1) refere-se ao método implementado, onde os cálculos das funções objetivas ocorrem para cada solução, conforme representado neste trabalho pelo Algoritmo 3. O parâmetro *algorithm* (linha 2) é o método de inicialização do algoritmo proveniente do Pymoo, a ser empregado no processo de otimização multiobjetivo, com as opções *NSGA2* e *AGEMOEA* selecionadas para este estudo. Por fim, o parâmetro *termination* (linha 3) é onde o critério de parada é definido com a função *get\_termination* do Pymoo, passando como parâmetro o critério a ser utilizado para encerrar o processo de otimização.

---

#### Algorithm 2 Multi Objective Search

---

**Input:** *data\_set*, *top\_n*, *pop\_size*, *metrics*, *mutation*, *crossover*, *type\_termination*

**Output:** Conjunto de soluções: *X*, valores das funções-objetivo por solução: *F*

```

1 problem ← FitnessEvaluation(top_n, data_set, metrics)
2 algorithm ← init_algorithm(pop_size, mutation, crossover)
3 termination ← get_termination(type_termination)
4 optimization_result ← minimize(problem, algorithm, termination)
5 X, F ← optimization_result.X, optimization_result.F
6 return X, F

```

---

O processo de busca se inicia com a geração de uma população inicial aleatória pelo

Pymoo, composta por várias soluções. O algoritmo 3, denominado *FitnessEvaluation*, assume a responsabilidade de calcular as funções objetivo para esse conjunto de soluções.

---

**Algorithm 3** Fitness Evaluation
 

---

**Input:** *population, top\_n, data\_set, metrics*

**Output:** Resultados das funções objetivos: *Out*

```

1 ratings ← getRatings(data_set)
2 objective_values ← []
3 foreach solution in population do
4   topn_scores ← get_top_n_score(data_set, solution, top_n)
5   user_metric_values ← []
6   foreach user, top_n_user in topn_scores.items() do
7     user_metrics ← []
8     foreach metric in metrics do
9       metric_value ← calculate_metric(top_n_user, ratings, metric)
10      user_metrics.append(metric_value)
11    end
12    user_metric_values.append(user_metrics)
13  end
14  user_avg_metrics ← mean(user_metric_values)
15  objective_values.append(user_avg_metrics)
16 end
17 return Out["F"] ← objective_values

```

---

Esse algoritmo recebe como parâmetros, a população, o valor de *top\_n*, que define o tamanho das listas de recomendação que serão geradas, e o conjunto de dados contendo o score de cada *feature*, juntamente com o *rating* correspondente para um par de usuário e item. Adicionalmente, são especificadas as métricas a serem utilizadas na avaliação.

A execução inicia-se com a obtenção dos *ratings* a partir do conjunto de dados de entrada (linha 1). Em seguida, ocorre a iteração em cada solução da população (linha 3). Para cada solução, são obtidas as *top\_n* recomendações para os usuários por meio do método *get\_top\_n\_score* (linha 4). Este método percorre o conjunto de dados, aplicando a hibridização ponderada normalizada a cada par de item e usuário, gerando um score resultante que é dividido restrito ao intervalo de 0 a 1 devido à normalização como mostrado na Equação 3.3.

$$\text{score\_normalizado} = \frac{(W1 \cdot F1) + (W2 \cdot F2) + \dots + (Wn \cdot Fn)}{n} \quad (3.3)$$

Baseado nesses scores é então aplicado um método de ordenação para obter a lista dos top-N items com maiores scores para serem candidatos a recomendação para cada usuário. Para ordenar os top-N items de cada usuário usamos o *nlargest* da biblioteca *heapq* (Python Software Foundation, 2022), ele é um método eficiente para encontrar os N maiores elementos de uma

lista, principalmente quando se está lidando com grandes conjuntos de dados. A eficiência do *nlargest* se deve ao uso de um *heap* binário. Um *heap* é uma estrutura de dados que permite que o elemento de maior prioridade seja acessado rapidamente.

Após gerado o top-N, é então aplicada as métricas escolhidas, EPC para avaliar a novidade e NDCG para avaliar a acurácia daquela recomendação candidata para cada usuário. Em seguida é aplicado a média das métricas para todos os usuários e armazenado. Esse processo ocorre para cada solução da população em questão, e ao final o algoritmo 3 retorna o conjunto de resultados das funções objetivos para cada solução.

Os algoritmos multiobjetivos utilizam esses resultados como base para verificar a qualidade de cada solução e a partir disso ocorrem os processos de seleção, mutação e *crossover* que definem as soluções que serão utilizadas para gerar a próxima geração.

O processo é repetido até concluir as rodadas, seguindo o critério de parada estabelecido, que neste trabalho é o tempo definido para cada experimento. Ao término da execução, é concluída a etapa de busca e gerado o conjunto de soluções não dominadas, representando a frente de Pareto.

### 3.2.1.3 Tomada de decisão

A seleção da solução a ser empregada é, portanto, fundamentada nesse conjunto de soluções ótimas de Pareto identificadas durante a fase de busca. A determinação da solução ideal deve ser orientada pelas particularidades individuais de cada experimento, levando em consideração as nuances específicas do problema e os objetivos distintos estabelecidos.

O Algoritmo 4 representa a implementação da tomada de decisão no *framework RecSysExp*. Inicialmente, o método de decisão da melhor solução recebe parâmetros essenciais: o conjunto de soluções da frente de Pareto (X), o conjunto de resultados das funções objetivas para cada solução da frente de Pareto (F), pesos a serem atribuídos a cada objetivo (*weights*), um parâmetro opcional para determinar se deseja utilizar algum dos métodos disponíveis no *framework (method)*, e *customFunction*, também opcional, para definir uma função específica diferente das oferecidas, a fim de realizar a tomada de decisão.

Dentre os métodos incorporados para realizar a tomada de decisão, será utilizada a estratégia conhecida como *Compromise Programming* (linhas 1 - 4), uma abordagem disponibilizada pelo *Pymoo*. Nesse contexto, a função responsável pela tomada de decisão utiliza a técnica de decomposição chamada *Achievement Scalarizing Function (ASF)* (NIKULIN; MIETTINEN; MÄKELÄ, 2012).

A ASF calcula um índice de compromisso para cada solução candidata, considerando a diferença entre suas funções objetivo e um ponto utópico que atua como uma referência ideal na busca. Além disso, a ASF leva em conta os pesos associados a cada objetivo, permitindo ao algoritmo de otimização ajustar o compromisso entre os objetivos de acordo com as preferências

do tomador de decisões.

Outra abordagem incorporada no *framework* é a escolha ponderada *weighted\_sum* (linhas 5 - 8), em que os valores das funções objetivas são combinados linearmente com pesos atribuídos a cada objetivo. Cada peso reflete a importância relativa de um objetivo específico no contexto do problema. A soma ponderada resulta em uma única função objetivo agregada, simplificando assim o problema multiobjetivo para um problema mono objetivo. Ao ajustar os pesos, os tomadores de decisões podem explorar diferentes compromissos entre os objetivos, permitindo uma personalização eficaz na busca pela solução ideal.

No **RecSysExp**, o tomador de decisão tem a opção de empregar uma função personalizada para orientar a seleção da solução mais adequada (linhas 9 - 11). Essa flexibilidade adicional permite adaptar a abordagem de decisão de acordo com as necessidades específicas de cada experimento, garantindo um ambiente mais personalizado e alinhado com os objetivos individuais.

---

#### Algorithm 4 Decide Best Solution

---

**Input:** X, F, weights, method = None, customFunction = None

**Output:** Solução resultante: X[index], index

---

```

1 if method == 'compromise_programming' then
2   |   decomp ← ASF()
3   |   index ← decomp(F, weights)
4 end
5 else if method == 'weighted_sum' then
6   |   values ← np.dot(F, weights)
7   |   index ← np.argmax(values)
8 end
9 else if callable(customFunction) then
10  |   index ← customFunction(F)
11 end
12 else
13  |   raise ValueError("Invalid decision method")
14 end
15 return X[index], index

```

---

### 3.2.2 Predição

Na fase de predição, após a escolha da solução durante o estágio de tomada de decisão, o sistema de recomendação utiliza essa solução para criar listas personalizadas de sugestões para os usuários. Essas predições baseiam-se nos critérios otimizados durante a etapa anterior, considerando múltiplos objetivos para fornecer recomendações mais refinadas e alinhadas com as preferências e necessidades individuais dos usuários. O processo de predição, portanto, representa a implementação prática das decisões tomadas durante a otimização.

As classes concretas, criadas para cada algoritmo integrado no *framework*, herdam os métodos de predição da classe abstrata *AbstractRecommender* e, conseqüentemente, implementam esses métodos. O Algoritmo 5, denominado *recommend*, desempenha a função de fornecer recomendações para todos os usuários. Essa funcionalidade é realizada por meio da invocação do método *get\_top\_n\_score*, responsável por gerar as listas de recomendação top-N para cada usuário.

O Algoritmo 6, intitulado *predict\_for\_user*, proporciona recomendações específicas para um usuário em particular. Sua implementação é semelhante à do método *recommend*. No entanto, ao retornar, ele disponibiliza os *topn\_scores* na posição correspondente ao usuário, resultando na lista de recomendação top-N específica para esse usuário.

---

**Algorithm 5** Recommend
 

---

**Input:** users, items, top\_n, solution, features

**Output:** Recomendações top\_n para o conjunto de usuários: topn\_scores

- 1 topn\_scores  $\leftarrow$  get\_top\_n\_score(users, items, solution, top\_n, features)
  - 2 **return** topn\_scores
- 

---

**Algorithm 6** Predict for user.
 

---

**Input:** user, items, top\_n, solution, features

**Output:** Recomendações top\_n para um usuário específico: topn\_scores[user]

- 1 topn\_scores  $\leftarrow$  get\_top\_n\_score(users, items, solution, top\_n, features)
  - 2 **return** topn\_scores[user]
- 

Esses algoritmos podem ser aplicados para obter listas de recomendações, dependendo dos objetivos estabelecidos no experimento. Posteriormente, realizamos uma avaliação das métricas novamente. Dado que a solução passou pelo processo de otimização, espera-se que essas listas estejam equilibradas em termos de acurácia e novidade.

Após a aplicação da solução escolhida e a obtenção das listas de recomendações por meio dos algoritmos específicos, seguida pela avaliação das métricas, o processo de recomendação multiobjetivo é concluído. Este ciclo de busca, tomada de decisão, aplicação da solução e avaliação contínua reflete a abordagem integrada do sistema de recomendação multiobjetivo, visando proporcionar sugestões personalizadas que equilibram acurácia e novidade.

## 4 Experimentos

Após a conclusão do desenvolvimento da inclusão dos novos métodos e métricas no *framework* **RecSysExp**, é essencial realizar uma avaliação abrangente para verificar o funcionamento e validar as implementações novas agregadas ao *framework*. Este capítulo apresenta os resultados dos experimentos computacionais realizados para demonstrar o uso do *framework*, começando com uma descrição das bases de dados utilizadas nos experimentos na Seção 4.1. Em seguida, na Seção 4.2, é detalhada a configuração do ambiente de testes. Os resultados são apresentados de forma organizada na Seção 4.3.

### 4.1 Base de dados

Foi utilizada a base de dados do *MovieLens* 1M previamente processada para atender às necessidades do trabalho de Fortes et al. (2021). Esta base inclui os dados originais do *MovieLens*, mas passou por um processo de pré-processamento. A base de dados do *MovieLens*, disponibilizada para pesquisas, engloba 1 milhão de avaliações, envolve a participação de 6000 usuários e abrange 4000 filmes distintos.

O processamento dos dados do *MovieLens* 1M realizado por Fortes et al. (2021) começa pela divisão desses dados em 5 *fold*s. Cada *fold* é então submetido a diferentes técnicas de hibridização. Na abordagem HR, hibridização ponderada, 13 algoritmos externos são aplicados ao conjunto de dados do *fold*. Cada algoritmo gera scores para o par item-usuário. Na abordagem STREAM, que utiliza *meta-features*, 59 algoritmos externos são aplicados ao conjunto de dados do *fold*. Os valores resultantes das *meta-features* e os *scores* dos algoritmos externos são combinados para formar os scores finais para cada par item-usuário.

Cada *fold*, em ambas as hibridizações, contém conjuntos de treino e teste, com informações como o par item-usuário, o *rating* e os *scores* de cada *feature*. Os valores de ratings foram normalizados para a faixa de 0 a 1 nessa base de dados.

### 4.2 Configurações dos experimentos

Todos os experimentos computacionais foram conduzidos em um computador equipado com um processador *Intel Core i7* de 2.30 GHz e 16 GB de RAM, executando o sistema operacional Ubuntu 22.04.3 LTS. O *framework* foi implementado utilizando *Python*.

Para realizar os experimentos foram realizadas algumas variações como pode ser visto na Tabela 4.1. Os experimentos foram conduzidos realizando variações nas meta heurísticas, na hibridização, nos operadores genéticos e no tamanho da população. No total, foram realizados

160 experimentos durante a fase de treinamento. Cada experimento foi limitado a uma duração de 1 hora, que serviu como critério de parada.

Tabela 4.1 – Variações Experimentos.

Parâmetros	Valores
Meta Heurística MO	NSGA2, AGEMOEA
Hibridização	STREAM, HR
Crossover	TwoPointCrossover(TPC) e SimulatedBinary-Crossover(SBC)
Mutação	PolynomialMutation(PM) e GaussianMutation(GM)
População	100, 500

Em todos os experimentos, foram consideradas uma tarefa de recomendação Top 5. As funções objetivas utilizadas foram EPC para novidade e NDCG para acurácia. Após a conclusão do processo de otimização, foi empregado o critério de decisão baseado em *compromise programming*, explicado na Subseção 4.3.2, com pesos de 0.5 para novidade e 0.5 para acurácia. Após a tomada de decisão sobre a melhor solução, foram utilizados os dados de teste da base de dados para aplicar a solução otimizada e gerar recomendações Top 5 para cada usuário, como detalhado na Subseção 3.2.2. Em seguida, foram combinadas as recomendações de cada fold e foram executadas novamente as métricas EPC e NDCG para avaliar o resultado final.

### 4.3 Resultados

Para validar a eficácia dos algoritmos multiobjetivos NSGA2 e AGEMOEA, incorporados ao *framework*, e avaliar o desempenho da métrica de novidade EPC, foram utilizadas ferramentas robustas de validação. Durante a etapa de otimização, a métrica de hipervolume foi utilizada para avaliar a qualidade das soluções de Pareto geradas pelos algoritmos NSGA2 e AGEMOEA, como será descrito na Subseção 4.3.1. Essa métrica proporciona uma visão abrangente da eficiência e diversidade das soluções ao longo das gerações.

Após a geração dessa frente, segue para a fase de decisão, na qual são exploradas as diferenças nas escolhas estratégicas implementadas no *framework*. As estratégias de *Compromise Programming* e *Weighted Sum* serão analisadas através de gráficos que destacam as escolhas dentro da frente de Pareto, como será mostrado na Subseção 4.3.2.

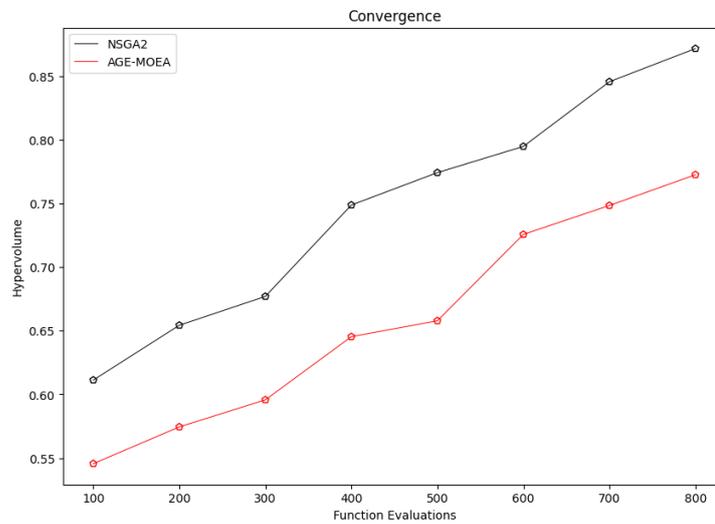
Posteriormente, na etapa de predição, é utilizado o intervalo de confiança como uma ferramenta estatística para avaliar a robustez da solução escolhida, proporcionando uma análise dos resultados obtidos durante todo o processo de otimização e tomada de decisão, como será detalhado na Subseção 4.3.3.

Ao empregar o intervalo de confiança durante a análise dos resultados, é possível determinar se existem diferenças estatisticamente significativas entre diferentes experimentos. Ao calcular

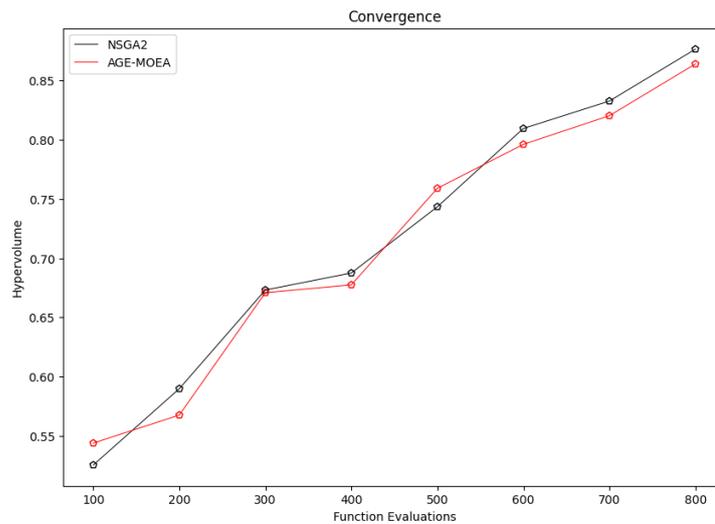
os intervalos de confiança para as métricas de interesse em cada experimento, a sobreposição desses intervalos é indicativa de possível equivalência estatística entre os grupos comparados. Se os intervalos não se sobrepuserem, isso sugere evidências de uma diferença significativa.

### 4.3.1 Otimização

O hipervolume é uma métrica amplamente utilizada em otimização multiobjetivo para avaliar a qualidade das soluções na fronteira de Pareto. Representando o volume do espaço dominado por essas soluções, o hipervolume mede a extensão da região que contém todas as soluções não dominadas em um espaço de objetivos multiobjetivo. Uma maior pontuação de hipervolume indica uma fronteira de Pareto mais extensa e diversificada.



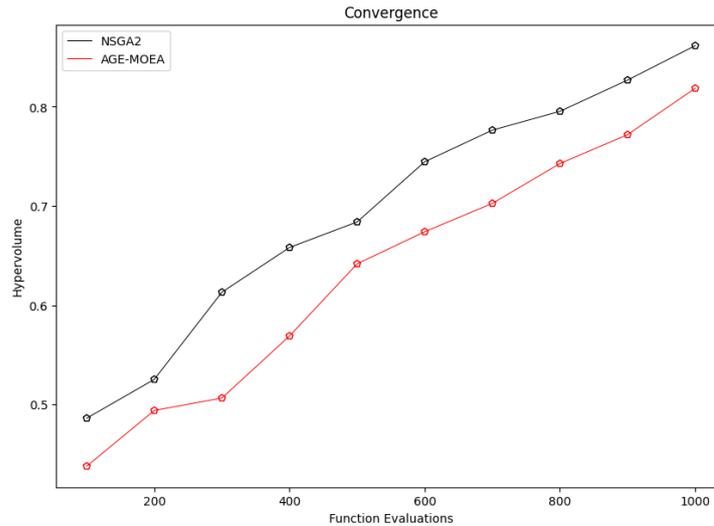
(a) GM SBC



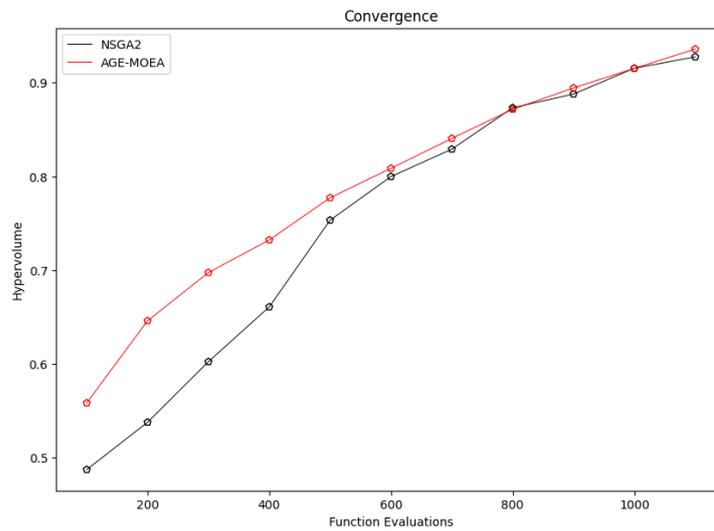
(b) PM SBC

Figura 4.1 – Evolução hipervolume STREAM - POP100.

Foi utilizado o hipervolume como métrica de avaliação para comparar os desempenhos dos métodos NSGA2 e AGEMOEA em um cenário experimental comum. Nas Figuras 4.1 e 4.2,



(a) GM SBC



(b) PM SBC

Figura 4.2 – Evolução hipervolume HR - POP100.

apresentamos visualizações da evolução do hipervolume em cada geração gerada por ambos os algoritmos. Essa análise fornece insights sobre como esses métodos exploram e expandem a fronteira de Pareto ao longo das iterações, permitindo uma compreensão mais aprofundada de seu comportamento em termos de eficiência e diversidade de soluções.

A biblioteca pymoo oferece uma funcionalidade nativa para salvar o histórico das gerações durante o processo de otimização. Aproveitamos essa característica incorporada no pymoo, que nos permite calcular o hipervolume para cada geração, registrando o histórico evolutivo. Dessa forma, foi possível obter a compreensão da evolução do hipervolume ao longo do tempo. Proporcionando uma visão sobre o desempenho comparativo dos métodos NSGA2 e AGEMOEA em nosso cenário experimental.

Ao interpretar o histórico do hipervolume gerado durante o processo de otimização, algumas observações relevantes emergem. Primeiramente, apesar de não ter atingido uma conver-

gência plena, evidenciando a necessidade de um tempo de execução mais prolongado, a tendência indicava uma direção positiva, sugerindo que a otimização estava se encaminhando para um estado mais favorável. Vale ressaltar que, como o principal objetivo dessa experimentação era validar as implementações, o critério de parada de 1 hora foi adotado para alcançar esse objetivo.

Outro ponto de destaque reside na constatação de que, de forma geral, o NSGA2 demonstrou um desempenho superior ao AGEMOEA ao longo das gerações. Esta superioridade foi claramente evidenciada pelo aumento mais significativo do hipervolume, refletindo uma exploração mais eficiente e expansão da fronteira de Pareto por parte do NSGA2. Ambos os algoritmos, no entanto, exibiram uma evolução positiva, conforme evidenciado pelo aumento constante do hipervolume a cada iteração.

Por fim, destaca-se a importância da qualidade da população inicial na rapidez com que o hipervolume evolui ao longo das gerações. Uma população inicial bem-calibrada pode acelerar significativamente o processo de melhoria, indicando a relevância crucial de escolhas estratégicas na configuração inicial dos algoritmos de otimização multiobjetivo.

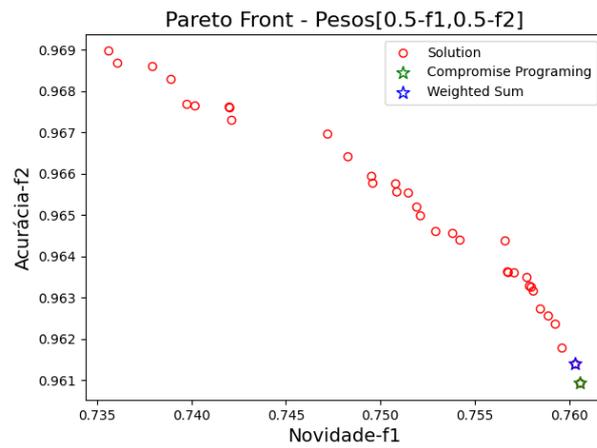
### 4.3.2 Decisão Melhor Solução

Conforme detalhado anteriormente, após a fase de otimização, é derivada uma frente de Pareto que reúne as soluções mais destacadas. Neste ponto, surge a necessidade de tomar uma decisão sobre qual solução é a mais adequada. A fim de elucidar as distinções ao variar os pesos atribuídos à seleção da melhor solução, bem como ao empregar as estratégias disponíveis no *framework* de escolha da melhor solução, como a *Compromise Programming* e a *Weighted Sum*, geramos gráficos ilustrativos das frentes de Pareto para alguns experimentos.

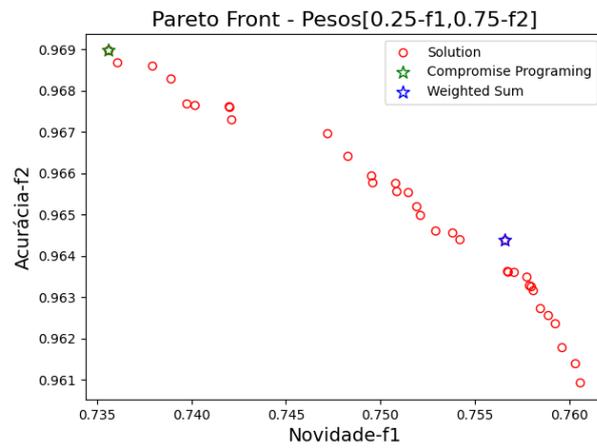
A Figura 4.3 apresenta as frentes de Pareto correspondentes ao experimento com as configurações:

- **Meta Heurística:** AGEMOEA
- **Hibridização:** HR
- **Crossover:** SimulatedBinaryCrossover
- **Mutação:** GaussianMutation
- **População:** 100

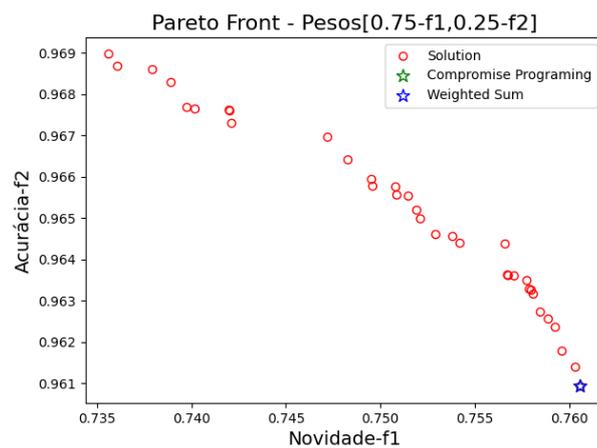
Cada ponto vermelho na representação gráfica indica uma solução pertencente à fronteira de Pareto final para o experimento em questão. A posição específica de cada ponto no gráfico reflete os resultados obtidos nas funções objetivas associadas a essa solução específica. Destaca-se, ainda, a presença de duas estrelas distintas: uma verde e outra azul.



(a) Frente de Pareto 0.5 0.5.



(b) Frente de Pareto 0.25 0.75.



(c) Frente de Pareto 0.75 0.25.

Figura 4.3 – Representação decisão da melhor solução com diferentes pesos.

A estrela verde representa a solução considerada como a melhor, adotando a estratégia de *Compromise Programming*. Essa abordagem leva em conta tanto os pesos atribuídos às funções objetivas quanto o ponto ideal no espaço de busca. Isso resulta em uma solução que busca um

equilíbrio entre os diferentes objetivos, levando em consideração a referência ao ponto ideal.

Por outro lado, a estrela azul denota a melhor solução escolhida utilizando a estratégia de *Weighted Sum*. Nessa estratégia, apenas os pesos associados às funções objetivas são levados em consideração, e uma ponderação é realizada para a tomada de decisão. Essa estratégia calcula uma combinação linear dos objetivos, refletindo uma preferência mais direta baseada nos pesos atribuídos a cada objetivo. A estrela azul, representando a melhor solução nessa perspectiva, destaca como a escolha da solução ótima pode variar significativamente dependendo da ênfase dada aos diferentes objetivos por meio da ponderação.

É importante ressaltar que a escolha entre *Compromise Programming* e *Weighted Sum* dependerá do propósito específico do experimento e das prioridades estabelecidas. Cada estratégia oferece uma perspectiva única na seleção de soluções ótimas, e a decisão sobre qual abordagem adotar será guiada pelas metas e objetivos específicos de cada estudo.

### 4.3.3 Predição

Após a determinação da melhor solução na etapa de decisão da otimização, o próximo passo é aplicar essa solução na fase de predição. Agregamos todas as predições provenientes de diferentes folds, e para cada configuração experimental, realizamos o cálculo das métricas de avaliação para cada usuário individualmente. Este processo é conduzido para garantir uma análise sensível às variações individuais. Posteriormente, calculamos a média das métricas obtidas para cada configuração, proporcionando uma visão geral do desempenho. Para avaliar a confiabilidade dessas métricas, empregamos o intervalo de confiança, fornecendo uma estimativa estatística da variação e da consistência dos resultados obtidos ao longo dos diferentes folds.

O intervalo de confiança é uma medida estatística que fornece uma faixa estimada na qual a verdadeira média de uma métrica pode estar, com um certo nível de confiança. No contexto da análise das métricas de novidade e acurácia, calculamos esses intervalos para avaliar a variabilidade das medidas e a precisão das estimativas.

Para a métrica de novidade, por exemplo, utilizamos o método de intervalo de confiança da distribuição t de *Student*. A fórmula envolve parâmetros como o nível de confiança (0.95), o número de amostras menos 1 e os valores da média e do erro padrão da métrica. O resultado é um intervalo que expressa com 95% de confiança a faixa na qual a verdadeira média da novidade pode residir.

O mesmo procedimento é aplicado à métrica de acurácia, usando as pontuações correspondentes. Em ambos os casos, a abordagem oferece uma maneira robusta de entender a variabilidade das métricas, auxiliando na interpretação dos resultados e fornecendo insights sobre a confiabilidade das medições realizadas.

Nas tabelas a seguir, pode-se observar os resultados obtidos, apresentando as informações derivadas dos intervalos de confiança calculados para as métricas de novidade e acurácia.

Tabela 4.2 – Intervalo de confiança novidade - HR - POP100.

Configuração	AGEMOEА			NSGA2		
	Intervalo Mínimo	Intervalo Máximo	Média	Intervalo Mínimo	Intervalo Máximo	Média
GM TPC	0.738918	0.745320	0.742119	0.739803	0.746235	0.743019
PM TPC	0.738560	0.744908	0.741734	0.738350	0.744716	0.741533
GM SBC	0.738976	0.745374	0.742175	0.738940	0.745315	0.742127
PM SBC	0.739113	0.745499	0.742306	0.739199	0.745573	0.742386

Tabela 4.3 – Intervalo de confiança novidade - HR - POP500.

Configuração	AGEMOEА			NSGA2		
	Intervalo Mínimo	Intervalo Máximo	Média	Intervalo Mínimo	Intervalo Máximo	Média
GM TPC	0.737651	0.744001	0.740826	0.737720	0.744079	0.740899
PM TPC	0.738062	0.744410	0.741236	0.737279	0.743641	0.740460
GM SBC	0.737241	0.743586	0.740414	0.736033	0.742401	0.739217
PM SBC	0.736736	0.743099	0.739918	0.737024	0.743395	0.740209

Tabela 4.4 – Intervalo de confiança novidade - STREAM - POP100.

Configuração	AGEMOEА			NSGA2		
	Intervalo Mínimo	Intervalo Máximo	Média	Intervalo Mínimo	Intervalo Máximo	Média
GM TPC	0.813739	0.820261	0.817000	0.803257	0.809419	0.806338
PM TPC	0.813665	0.820277	0.816971	0.793614	0.800036	0.796825
GM SBC	0.811809	0.818366	0.815088	0.804742	0.811122	0.807932
PM SBC	0.818356	0.825046	0.821701	0.801124	0.807535	0.804330

Ao examinar os resultados nas tabelas para as condições de população de 100 e 500, tanto para os algoritmos NSGA2 quanto para AGEMOEА, destaca-se uma conclusão relevante sobre a influência das bases de dados HR e STREAM na novidade. A análise dos intervalos de confiança revela uma ausência de sobreposição entre os intervalos correspondentes para ambas as bases de dados, sugerindo com confiança estatística que a base de dados com hibridização STREAM é estatisticamente superior à base de dados HR em relação à novidade. As estimativas pontuais indicam uma tendência mais elevada nos resultados provenientes da base de dados STREAM, enquanto a base de dados HR apresenta valores inferiores.

Esse resultado era esperado, considerando que a base de dados que utiliza a hibridização STREAM possui um maior número de *features*, incluindo a influência de *meta-features* projetadas para aprimorar os resultados. Consequentemente, isso reforça a eficácia esperada dos algoritmos NSGA2 e AGEMOEА, validando sua capacidade de operar de acordo com as expectativas estabelecidas.

Tabela 4.5 – Intervalo de confiança novidade - STREAM - POP500.

Configuração	AGEMOEА			NSGA2		
	Intervalo Mínimo	Intervalo Máximo	Média	Intervalo Mínimo	Intervalo Máximo	Média
GM TPC	0.787181	0.793764	0.790472	0.797803	0.803940	0.800872
PM TPC	0.787753	0.794412	0.791083	0.795696	0.801786	0.798741
GM SBC	0.783307	0.789750	0.786528	0.794050	0.800285	0.797168
PM SBC	0.782698	0.789141	0.785919	0.801474	0.807523	0.804499

Ao analisar o intervalo de confiança para a novidade nos métodos NSGA2 e AGEMOEA, observa-se que não há diferenças estatisticamente significativas entre esses dois algoritmos. Os intervalos de confiança para ambas as abordagens apresentam sobreposições sutis, indicando uma consistência nas estimativas pontuais e uma ausência de divergências estatísticas. Esses resultados sugerem que, para a métrica de novidade, NSGA2 e AGEMOEA demonstraram desempenhos semelhantes, não sendo possível afirmar com confiança a superioridade estatística de um sobre o outro. Essa uniformidade nos resultados pode ter implicações práticas, indicando que, pelo menos para a novidade, ambos os métodos se mostraram comparáveis em termos de eficácia.

Analisando as métricas de novidade de forma abrangente para o EPC, observamos resultados consistentes e promissores em todas as configurações consideradas. Independentemente da meta-heurística e das diferentes combinações de parâmetros, as médias de novidade mantiveram-se em níveis satisfatórios, evidenciando a capacidade do modelo em proporcionar recomendações inovadoras. Essa estabilidade nas métricas de novidade ao longo das diversas configurações reforça a confiabilidade dos métodos inclusos no *framework*, destacando sua capacidade de explorar eficientemente o espaço de busca e fornecer soluções equilibradas e diversas.

Tabela 4.6 – Intervalo de Confiança Acurácia - HR - POP100.

Configuração	AGEMOEA			NSGA2		
	Intervalo Mínimo	Intervalo Máximo	Média	Intervalo Mínimo	Intervalo Máximo	Média
GM TPC	0.960295	0.963822	0.962058	0.960138	0.963669	0.961903
PM TPC	0.959331	0.962837	0.961084	0.958599	0.962207	0.960403
GM SBC	0.959783	0.963337	0.961560	0.959289	0.962871	0.961080
PM SBC	0.958609	0.962212	0.960410	0.958227	0.961878	0.960052

Tabela 4.7 – Intervalo de Confiança Acurácia - HR - POP500.

Configuração	AGEMOEA			NSGA2		
	Intervalo Mínimo	Intervalo Máximo	Média	Intervalo Mínimo	Intervalo Máximo	Média
GM TPC	0.958889	0.962469	0.960679	0.959457	0.963000	0.961229
PM TPC	0.958506	0.962069	0.960287	0.959008	0.962563	0.960786
GM SBC	0.958527	0.962140	0.960334	0.958511	0.962121	0.960316
PM SBC	0.958292	0.961906	0.960099	0.958678	0.962277	0.960478

Tabela 4.8 – Intervalo de confiança acurácia - STREAM - POP100.

Configuração	AGEMOEA			NSGA2		
	Intervalo Mínimo	Intervalo Máximo	Média	Intervalo Mínimo	Intervalo Máximo	Média
GM TPC	0.931957	0.937079	0.934518	0.942165	0.946515	0.944340
PM TPC	0.931593	0.936759	0.934176	0.938682	0.943291	0.940986
GM SBC	0.933409	0.938366	0.935888	0.940347	0.944792	0.942570
PM SBC	0.926313	0.931843	0.929078	0.940365	0.944831	0.942598

Ao avaliar a acurácia utilizando a base de dados que incorpora o STREAM, notou-se que o método NSGA2 apresentou um desempenho ligeiramente superior em comparação ao AGEMOEA. Esta observação é respaldada pela inexistência de sobreposição nos intervalos de confiança médios entre os dois métodos, indicando uma diferença estatística não negligenciável.

Tabela 4.9 – Intervalo de confiança acurácia - STREAM - POP500.

Configuração	AGEMOEA			NSGA2		
	Intervalo Mínimo	Intervalo Máximo	Média	Intervalo Mínimo	Intervalo Máximo	Média
GM TPC	0.943622	0.947922	0.945772	0.959457	0.963000	0.961229
PM TPC	0.939819	0.944497	0.942158	0.959008	0.962563	0.960786
GM SBC	0.944630	0.948864	0.946747	0.958511	0.962121	0.960316
PM SBC	0.939601	0.944228	0.941914	0.958678	0.962277	0.960478

Os valores médios para o NSGA2 estão marginalmente acima daqueles obtidos pelo AGEMOEA, denotando uma consistência nos resultados. Embora os intervalos não se sobreponham, é importante notar que as discrepâncias entre as médias são sutis, sugerindo uma proximidade significativa em termos de desempenho. Essa proximidade nas estimativas ressalta que, embora o NSGA2 tenha demonstrado uma superioridade estatística, a diferença em valores é mínima e, portanto, ambos os métodos apresentam desempenhos bastante similares em relação à acurácia.

De maneira abrangente, a análise da acurácia revelou a ausência de discrepâncias substanciais que comprovem melhorias estatisticamente significativas em diversos cenários experimentais. Essa constatação é consistente ao considerar as diferentes combinações de bases de dados (HR e STREAM), tamanhos de população (POP100 e POP500), configurações, assim como os algoritmos AGEMOEA e NSGA2. A observação de intervalos de confiança para cada cenário não evidenciou diferenças marcantes entre os métodos, indicando que, em termos de acurácia, as variações nos parâmetros experimentais não resultaram em diferenças estatisticamente significativas. Esse panorama sugere uma estabilidade relativa nos desempenhos dos algoritmos AGEMOEA e NSGA2, independentemente das diferentes condições experimentais avaliadas.

Os consistentes resultados de acurácia, evidenciados pelas médias de NDCG acima de 0.9 em todos os cenários experimentais testados para ambas as estratégias de otimização, AGEMOEA e NSGA2, indicam que as recomendações geradas apresentaram elevada precisão. O NDCG, métrica sensível à ordem das recomendações, reforça que as sugestões bem posicionadas foram um destaque. A consistência desses resultados fortalece a confiabilidade estatística das abordagens implementadas na geração de recomendações precisas.

## 5 Conclusões

Neste trabalho, foi proposto a expansão do *framework* **RecSysExp** por meio da inclusão de métodos multiobjetivo, visando ampliar a cobertura de experimentos realizáveis com o **RecSysExp**. Integramos métodos multiobjetivos notáveis, como NSGA2 e AGEMOEA, ao *framework*. Para possibilitar experimentos multiobjetivo, também introduzimos a métrica EPC, uma medida de novidade.

O **RecSysExp**, um *framework* generalista, busca tornar-se mais abrangente em diversos cenários de recomendação, promovendo a inclusão de uma métrica específica de novidade e adaptando-se para lidar com métodos multiobjetivo. Ao integrar a otimização multiobjetivo em sistemas de recomendação, delineamos um processo composto por duas fases distintas: otimização e predição. A busca por soluções não dominadas ocorre inicialmente, seguida pela tomada de decisão para escolher uma solução, culminando no processo de predição e geração de listas de recomendações para os usuários com base na solução escolhida.

Os resultados obtidos nos experimentos revelaram um desempenho notável nas estratégias de otimização AGEMOEA e NSGA2. Tanto a acurácia, com médias consistentemente elevadas, quanto a novidade, mantendo-se em intervalos desejáveis, indicam a conquista de um equilíbrio eficaz entre a precisão das recomendações, que reflete a assertividade das recomendações em relação aos interesses dos usuários, e a diversificação do conteúdo sugerido, assegurando a apresentação de sugestões inovadoras. O equilíbrio alcançado ressalta a capacidade das estratégias implementadas em fornecer recomendações precisas e diversificadas.

Além disso, a avaliação por meio do hipervolume permitiu verificar o correto funcionamento dos algoritmos, proporcionando uma visualização da evolução esperada do hipervolume ao longo dos experimentos. Esses resultados consolidam a eficácia e a aplicabilidade do *framework* expandido, destacando sua utilidade na condução de experimentos multiobjetivo em sistemas de recomendação.

A execução de experimentos de sistema de recomendação multiobjetivo por meio da funcionalidade incluída no *framework* apresenta algumas limitações. A principal delas está relacionada à rigidez do formato dos dados de entrada. É necessário que os dados estejam estruturados de maneira específica, onde para cada par de item e usuário, seja fornecido o *rating* associado, seguido pelos *scores* correspondentes de cada feature. Essa exigência pode ser restritiva, especialmente se os conjuntos de dados disponíveis não estiverem originalmente organizados nesse formato particular, demandando um esforço adicional na preparação dos dados para se adequarem à entrada necessária.

Além disso, uma limitação importante do *framework* está relacionada à escassez de métricas de avaliação disponíveis. Embora tenha sido incorporada uma métrica que avalia a

novidade das recomendações, ainda falta a inclusão de métricas que abordem aspectos como diversidade, serendipidade e fairness. Essa ausência pode representar uma restrição dos cenários experimentais que podem ser abordados pelo *framework*.

Dito isso, como perspectiva futura, pode-se considerar a integração de métricas abrangentes que incluam tanto a diversidade quanto critérios de *fairness* em nosso *framework*. Essa evolução possibilitará a execução simultânea de algoritmos de otimização envolvendo mais de dois objetivos, ampliando significativamente a capacidade do sistema. Além disso, pode-se também expandir os métodos de recomendação dentro do *framework*, proporcionando uma abordagem mais abrangente para atender às necessidades em constante evolução. Garantindo que ele esteja alinhado com as demandas cada vez mais complexas e diversificadas do cenário atual de Sistemas de Recomendação.

# Referências

- ADOMAVICIUS, G.; ZHANG, J. Impact of data characteristics on recommender systems performance. *ACM Trans. Manag. Inf. Syst.*, v. 3, p. 3:1–3:17, 2012. Disponível em: <<https://api.semanticscholar.org/CorpusID:11146681>>.
- AGGARWAL, C. C. *Recommender Systems - The Textbook*. [S.l.]: Springer, 2016. 1-498 p. ISBN 978-3-319-29659-3.
- ANELLI, V. W.; BELLOGÍN, A.; FERRARA, A.; MALITESTA, D.; MERRA, F. A.; POMO, C.; DONINI, F. M.; NOIA, T. D. Elliot: a comprehensive and rigorous framework for reproducible recommender systems evaluation. *CoRR*, abs/2103.02590, 2021. Disponível em: <<https://arxiv.org/abs/2103.02590>>.
- BAO, X.; BERGMAN, L.; THOMPSON, R. Stacking recommendation engines with additional meta-features. In: . [S.l.: s.n.], 2009. p. 109–116.
- BLANK, J.; DEB, K. Pymoo: Multi-objective optimization in python. *IEEE Access*, PP, p. 1–1, 04 2020.
- BOBADILLA, J.; ORTEGA, F.; HERNANDO, A.; GUTIÉRREZ, A. Recommender systems survey. *Knowledge-Based Systems*, v. 46, p. 109–132, 07 2013.
- BRUCKNER, C. H. P. *FRAMEWORK PARA EXTRAÇÃO DE META-FEATURES PARA BASES DE DADOS DE SISTEMAS DE RECOMENDAÇÃO*. 45 f. Monografia (Monografia) — Universidade Federal de Ouro Preto, Instituto de Ciências Exatas e Biológicas, Departamento de Computação, Minas Gerais, 2017.
- BURKE, R. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, v. 12, 11 2002.
- ÇANO, E.; MORISIO, M. Hybrid recommender systems: A systematic literature review. *CoRR*, abs/1901.03888, 2019. Disponível em: <<http://arxiv.org/abs/1901.03888>>.
- COELLO, C.; VELDHUIZEN, D.; LAMONT, G. *Evolutionary Algorithms for Solving Multi-Objective Problems Second Edition*. [S.l.: s.n.], 2007. ISBN 978-0-387-33254-3.
- DEB, K. Multi-objective optimisation using evolutionary algorithms: An introduction. In: \_\_\_\_\_. [S.l.: s.n.], 2011. p. 3–34. ISBN 978-0-85729-617-7.
- DEB, K.; PRATAP, A.; AGARWAL, S.; MEYARIVAN, T. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, v. 6, n. 2, p. 182–197, 2002.
- DIAS, M. D. P. G. *XPERIMENTOR: UM FRAMEWORK PARA O GERENCIAMENTO DE EXECUÇÃO DE EXPERIMENTOS COMPUTACIONAIS*. 45 f. Monografia (Monografia) — Universidade Federal de Ouro Preto, Instituto de Ciências Exatas e Biológicas, Departamento de Computação, Minas Gerais, 2019.
- FERREIRA, F. G. D. C. Estudo comparativo de modelos e técnicas para otimização de portfólios com restrição de cardinalidade. 2018.

FORTES, R. S. Enhancing the multi-objective recommendation from three new perspectives: data characterization, risk-sensitiveness, and prioritization of the objectives. 2022.

FORTES, R. S.; SOUSA, D. X. de; COELHO, D. G.; LACERDA, A. M.; GONÇALVES, M. A. Individualized extreme dominance (inded): A new preference-based method for multi-objective recommender systems. *Inf. Sci.*, v. 572, p. 558–573, 2021. Disponível em: <<https://doi.org/10.1016/j.ins.2021.05.037>>.

GUNAWARDANA, A.; SHANI, G. A survey of accuracy evaluation metrics of recommendation tasks. *J. Mach. Learn. Res.*, JMLR.org, v. 10, p. 2935–2962, dec 2009. ISSN 1532-4435.

HERLOCKER, J. L.; KONSTAN, J. A.; TERVEEN, L. G.; RIEDL, J. T. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, Association for Computing Machinery, New York, NY, USA, v. 22, n. 1, p. 5–53, jan 2004. ISSN 1046-8188. Disponível em: <<https://doi.org/10.1145/963770.963772>>.

JANNACH, D.; ABDOLLAHOPOURI, H. A survey on multi-objective recommender systems. *Frontiers in Big Data*, v. 6, 2023. ISSN 2624-909X. Disponível em: <<https://www.frontiersin.org/articles/10.3389/fdata.2023.1157899>>.

JÄRVELIN, K.; KEKÄLÄINEN, J. Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst.*, Association for Computing Machinery, New York, NY, USA, v. 20, n. 4, p. 422–446, oct 2002. ISSN 1046-8188. Disponível em: <<https://doi.org/10.1145/582415.582418>>.

LOPS, P.; GEMMIS, M. de; SEMERARO, G. Content-based recommender systems: State of the art and trends. In: \_\_\_\_\_. [S.l.: s.n.], 2011. p. 73–105. ISBN 978-0-387-85819-7.

LÜCKEN, C.; BARÁN, B.; BRIZUELA, C. A survey on multi-objective evolutionary algorithms for many-objective problems. *Computational Optimization and Applications*, v. 58, n. 3, p. 707–756, July 2014. Disponível em: <<https://ideas.repec.org/a/spr/cooap/v58y2014i3p707-756.html>>.

MCNEE, S.; RIEDL, J.; KONSTAN, J. Being accurate is not enough: How accuracy metrics have hurt recommender systems. In: . [S.l.: s.n.], 2006. p. 1097–1101.

MCNEE, S. M.; RIEDL, J.; KONSTAN, J. A. Being accurate is not enough: How accuracy metrics have hurt recommender systems. In: *CHI '06 Extended Abstracts on Human Factors in Computing Systems*. New York, NY, USA: Association for Computing Machinery, 2006. (CHI EA '06), p. 1097–1101. ISBN 1595932984. Disponível em: <<https://doi.org/10.1145/1125451.1125659>>.

NABIZADEH, A. H.; RAFSANJANI, N.; SALIM, N.; AGHDAM, A. R.; FARD, K. Recommendation systems: a review. 05 2013.

NATALI, L. M. S. *RecSysExp: Um framework de alto nível de abstração para implementação e validação de sistemas de recomendação*. 82 f. Monografia (Monografia) — Universidade Federal de Ouro Preto, Instituto de Ciências Exatas e Biológicas, Departamento de Computação, Minas Gerais, 2023.

NIKULIN, Y.; MIETTINEN, K.; MÄKELÄ, M. M. A new achievement scalarizing function based on parameterization in multiobjective optimization. *OR Spectrum*, v. 34, n. 1, p. 69–87, 2012. Disponível em: <<https://doi.org/10.1007/s00291-010-0224-1>>.

PANICHELLA, A. An adaptive evolutionary algorithm based on non-euclidean geometry for many-objective optimization. In: . [S.l.: s.n.], 2019.

Python Software Foundation. *heapq – Heap queue algorithm*. 2022. Disponível em: <<https://docs.python.org/3/library/heapq.html>>.

RASHID, A.; ALBERT, I.; COSLEY, D.; LAM, S.; MCNEE, S.; KONSTAN, J.; RIEDL, J. Getting to know you: Learning new user preferences in recommender systems. *International Conference on Intelligent User Interfaces, Proceedings IUI*, 02 2002.

RIBEIRO, M. T.; ZIVIANI, N.; MOURA, E. S. D.; HATA, I.; LACERDA, A.; VELOSO, A. Multiobjective pareto-efficient approaches for recommender systems. *ACM Trans. Intell. Syst. Technol.*, Association for Computing Machinery, New York, NY, USA, v. 5, n. 4, dec 2015. ISSN 2157-6904. Disponível em: <<https://doi.org/10.1145/2629350>>.

RICCI, F.; ROKACH, L.; SHAPIRA, B. Introduction to recommender systems handbook. In: \_\_\_\_\_. *Recommender Systems Handbook*. Boston, MA: Springer US, 2011. p. 1–35. ISBN 978-0-387-85820-3. Disponível em: <[https://doi.org/10.1007/978-0-387-85820-3\\_1](https://doi.org/10.1007/978-0-387-85820-3_1)>.

SCHAFFER, J. B.; FRANKOWSKI, D.; HERLOCKER, J.; SEN, S. Collaborative filtering recommender systems. In: . [S.l.: s.n.], 2007.

SHAH, K.; SALUNKE, A.; DONGARE, S.; ANTALA, K. Recommender systems: An overview of different approaches to recommendations. In: *2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS)*. [S.l.: s.n.], 2017. p. 1–4.

SOUZA, A. N. D. P. E. *UM FRAMEWORK PARA O CÁLCULO DE MÉTRICAS DE CARACTERIZAÇÃO DE DADOS DE FILTRAGEM COLABORATIVA*. 60 f. Monografia (Monografia) — Universidade Federal de Ouro Preto, Instituto de Ciências Exatas e Biológicas, Departamento de Computação, Minas Gerais, 2014.

SUN, Z.; FANG, H.; YANG, J.; QU, X.; LIU, H.; YU, D.; ONG, Y.-S.; ZHANG, J. *DaisyRec 2.0: Benchmarking Recommendation for Rigorous Evaluation*. 2022.

SUN, Z.; GUO, Q.; YANG, J.; FANG, H.; GUO, G.; ZHANG, J.; BURKE, R. Research commentary on recommendations with side information: A survey and research directions. *CoRR*, abs/1909.12807, 2019. Disponível em: <<http://arxiv.org/abs/1909.12807>>.

VARGAS, S.; CASTELLS, P. Rank and relevance in novelty and diversity metrics for recommender systems. In: . [S.l.: s.n.], 2011. p. 109–116.

VILLA, C.; LABAYRADE, R. Energy efficiency vs subjective comfort: A multiobjective optimisation method under uncertainty. *Proc. of the Building Simulation 2011 Conf., ...*, 01 2011.

WANG, Y.; WANG, L.; LI, Y.; HE, D.; LIU, T.; CHEN, W. A theoretical analysis of NDCG type ranking measures. *CoRR*, abs/1304.6480, 2013. Disponível em: <<http://arxiv.org/abs/1304.6480>>.

ZHANG, M.; HURLEY, N. Avoiding monotony: Improving the diversity of recommendation lists. In: *Proceedings of the 2008 ACM Conference on Recommender Systems*. New York, NY, USA: Association for Computing Machinery, 2008. (RecSys '08), p. 123–130. ISBN 9781605580937. Disponível em: <<https://doi.org/10.1145/1454008.1454030>>.

ZHANG, Q.; LI, H. Moea/d: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, v. 11, n. 6, p. 712–731, 2007.

ZHENG, Y.; WANG, D. X. Multi-objective recommendations: A tutorial. *CoRR*, abs/2108.06367, 2021. Disponível em: <<https://arxiv.org/abs/2108.06367>>.

ZHENG, Y.; WANG, D. X. A survey of recommender systems with multi-objective optimization. *Neurocomputing*, v. 474, p. 141–153, 2022. ISSN 0925-2312. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S0925231221017185>>.